Departamento de Telemática

PROYECTO FIN DE CARRERA

APLICACIONES MOVILES NATIVAS CON CONSUMO DE APIS ONLINE, ESTUDIO COMPARADO CON APLICACIONES WEB MOVILES EN IOS Y ANDROID Y CASO PRACTICO DE "NATIVE CLIENT" PARA WORDPRESS

Autor: Patricia Rincón Andrés

Tutor: Luis Ángel Galindo Sánchez

Título: APLICACIONES MOVILES NATIVAS CON CONSUMO DE APIS ONLINE, ESTUDIO COMPARADO CON APLICACIONES WEB MOVILES EN IOS Y ANDROID Y CASO PRACTICO DE "NATIVE CLIENT" PARA WORDPRESS

Autor: Patricia Rincón Andrés
Director: Luis Ángel Galindo Sánchez

EL TRIBUNAL

Presidente: _____

Vocal: _____

Secretario: _____

Realizado el acto de defensa y lectura del Proyecto Fin de Carrera el día X de Julio de 2012 en Leganés, en la Escuela Politécnica Superior de la Universidad Carlos III de Madrid, acuerda otorgarle la CALIFICACIÓN de

VOCAL

SECRETARIO PRESIDENTE

Resumen

La aparición de los teléfonos inteligentes ha supuesto una auténtica revolución sustituyendo el clásico concepto de dispositivo móvil por el de pequeñas computadoras de bolsillo. Esta nueva filosofía abre un mundo de posibilidades al usuario, dándole la libertad de tener una gran diversidad de aplicaciones en su dispositivo pero sobretodo permitiendo la comunicación y el acceso a la información en cualquier momento y desde cualquier lugar. Por ello hoy en día un gran número de compañías, sobretodo del sector tecnológico se ven impulsadas a tener su propia presencia en este mundo, desarrollando en muchos casos sus propias aplicaciones móviles.

Este proyecto se centra en el diseño e implementación de una aplicación móvil que de acceso al contenido de una página web. Esta página tiene como sistema de gestión de contenido Wordpress por lo que veremos el caso de un cliente para esta plataforma. Para comunicarnos con ella estudiaremos distintas formatos de intercambio de datos como son REST, XML-RPC y JSON, con el consumo de APIs online. Las aplicaciones desarrolladas serán un cliente nativo Android y otro en iOS, utilizando para este segundo la plataforma Titanium, basada en el desarrollo de aplicaciones móviles a través de tecnologías web (HTML, JavaScript y CSS). A la vez se hará un estudio comparativo entre estas aplicaciones nativas y el caso de aplicaciones web desarrolladas con HTML5 para el mismo objetivo pudiendo ver así claramente sus ventajas y desventajas tanto para el desarrollador como para el usuario final.

Índice de contenidos

Índice de figuras

Índice de tablas

En este punto se introduce el presente proyecto exponiendo que ha sido lo que ha motivado su realización así como los objetivos que se buscan en el mismo. Posteriormente se hace un pequeño recorrido por el resto de puntos explicando brevemente lo que se tratará en cada uno de ellos.

1.1 Motivación

Desde la aparición de los dispositivos móviles en torno a los años 80 se ha producido una evolución espectacular, surgieron por la idea de conseguir teléfonos sin cable, buscando principalmente en un origen ir reduciendo su tamaño y aumentando la capacidad de sus baterías, poco a poco se fue añadiendo funcionalidad como los mensajes de texto y algunos sencillos juegos. Poco a poco han ido ampliando estas funciones convergiendo cada vez mas hacia las computadoras, permitiendo instalar nuestras propias aplicaciones, haciendo de reproductores de música, gps, conexión a internet hasta una cada vez mayor interacción con el usuario, acelerómetro, realidad aumentada, etc. Por lo tanto no dejan de ser pequeños ordenadores con sus limitaciones pero a la vez brindando la posibilidad de tener nuestro pequeño computador en el bolsillo. Y es que hoy en día la posibilidad de acceder a la información en cualquier momento y desde cualquier lugar ha cobrado gran relevancia

La evolución constante de las tecnologías va obligando a la sociedad en todos los ámbitos a adaptarse constantemente a la nueva situación y al cada vez mayor alcance que permiten. Si tenemos en cuenta que pronto el número de usuarios navegando desde móviles superará al de usuarios navegando desde PC's es muy importante buscar la forma de seguir llegando a ellos a través de esta nueva vía. Y a pesar de que estos dispositivos tienen navegadores, conviene buscar una forma de adaptarse a las pequeñas pantallas. Para ello y por la facilidad de distribución es una buena opción crear nuestras propias aplicaciones móviles, adaptándolas a las características de los mismos y adaptando nuestra interfaz a estas dimensiones haciéndola lo más atractiva y usable posible.

Los principales sistemas operativos que han revolucionado el mercado de los smartphones son Android y iOS por lo que no parece mala idea centrarnos a priori en los mismos a la hora de desarrollar nuestras aplicaciones. Ambos cuentan con su plataforma de distribución que facilitará muchísimo llegar a todo el mundo, en el caso de Android tenemos el Google Play y en el caso de iOS el App Store.

Dado que en la actualidad es habitual que tanto empresas como personas físicas tengan su presencia en la web bien sea a través de webs corporativas, así como blogs o diversos tipos de portales es bueno pensar en adaptarse a los nuevos medios de acceso a la información que pretenden ofrecer. Así por ejemplo Wordpress es uno de lo más populares gestores de contenido en la actualidad con un muy elevado número de usuarios y con una gran diversidad de tipos de sitios web realizados con el mismo.

Por tanto parece de gran interés conseguir de una forma sencilla y lo más ágil posible acceder a toda la información de un gestor de contenido como Wordpress y poder adaptarla a un dispositivo reducido bien sea un dispositivo móvil o una Tablet. Esto supone una gran ventaja a su vez que es tener en todo momento actualizada la aplicación móvil con el contenido de la web.

Por tanto se conseguirá un cliente Wordpress para móviles que además será complementado con el acceso a otra API que devolverá los formularios adecuados para la aplicación (en este caso Contacto y Únete) y tendrá acceso directo a diversas redes sociales tanto para consultar el perfil del propietario de la web en las mismas así como para compartir los diferentes contenidos de Wordpress a los que se podrá acceder desde la aplicación.

1.2 Objetivos

Diseño e implementación de una aplicación móvil capaz de comunicarse con Wordpress y con Codeeta(software para creación y exportación de widgets) para obtener la información disponible en la página web y presentársela al usuario de la forma más atractiva y con la mayor usabilidad posible. La aplicación, dadas las limitaciones de este tipo de dispositivos, se centrará en dar acceso únicamente a una serie de contenidos considerados más relevantes para la misma. Se quiere sincronizar así el contenido de la web y la aplicación móvil por la comodidad que supone dado que hay mucho contenido dinámico que cambiará diariamente.

Esta aplicación ha sido desarrollada para Movilforum | Programa de partners de telefónica, desde la aplicación podremos:

- Acceder al contenido de presentación, donde se dan a conocer y exponen los beneficios de este programa.
- Podremos ver un catálogo con diversas soluciones propuestas y casos de éxitos.
- Consultar noticias y eventos
- Visualización del blog
- Podremos acceder tanto a un formulario de contacto para cualquier consulta como unirnos desde nuestros dispositivos a este programa
- Comunicación directa con redes sociales, desde consultar el perfil de Movilforum en las mismas hasta hacer recomendaciones en nuestros propios perfiles.

Para conseguir llegar a tal meta deberemos ir cumpliendo una serie de subobjetivos:

1. Instalación y toma de contacto con Wordpress
2. Toma de contacto con Codeeta
3. Investigar la forma de comunicarse con Wordpress y Codeeta, documentarse para ello sobre los diferentes formatos de intercambio de datos y valorando las distintas posibilidades.

4. Estudio sobre la plataforma Android, conociendo su filosofía y la forma de desarrollar para la misma. Eclipse
5. Estudio sobre la plataforma iOS, conociendo su filosofía y la forma de desarrollar para la misma. Titanium
6. Se debe pensar en dar persistencia a la aplicación dotándola de comportamiento offline

Una vez llegado a este punto estaremos preparados para comenzar con la aplicación

7. Diseño de la aplicación
8. Desarrollo de la aplicación
9. Pruebas

1.3 Estructura de la memoria

En este punto se define como está organizada esta memoria, describiendo lo que se trata en cada uno de los puntos.

1. Introducción

Es el punto en el que nos encontramos, en el cual tratamos la motivación que ha llevado a realizar este proyecto así como los objetivos que se pretenden y cómo esta estructurada la memoria.

2. Estado del arte

En este punto se analiza la situación actual de diversas tecnologías, plataformas, formatos de comunicación, etc. viendo cuales serán las óptimas para el desarrollo de este proyecto

- Veremos los diferentes Sistemas operativos para smartphones haciendo especial incapié en Android y iOS.
- Seguimos con un acercamiento al sistema de gestión de contenido Wordpress.
- Veremos diferentes formatos con lo que podríamos comunicarnos con wordpress comparándolos para encontrar la mejor opción posible.
- Eclipse y Titanium como entornos para el desarrollo de la aplicación en Android e iOS respectivamente.
- Finalmente haremos una comparativa entre las aplicaciones móviles nativas y las aplicaciones web viendo sus ventajas e inconvenientes

3. Análisis del sistema

Aquí analizaremos dados los objetivos que se buscan la funcionalidad que tendrá la aplicación, viendo los requisitos de software y de usuario necesarios antes de comenzar a diseñar e implementar la aplicación.

4. Diseño de la aplicación
   En este punto se diseña la aplicación, en base a los objetivos que se pretenden se busca un diseño que cumpla de la forma más eficiente con su propósito. Se buscara una aplicación usable y fácil de manejar a la vez que visualmente atractiva. Aquí se tratará el diseño a todos los niveles, desde las comunicaciones y la obtención de datos hasta su almacenamiento para posteriormente mostrarlos mediante la interfaz de usuario.

5. Implementación
   En este punto se procede a explicar el desarrollo de la aplicación, viendo que dificultades se han encontrado a lo largo del mismo y que decisiones se han ido tomando para solventarlas. Finalmente se prueba para comprobar que cumple con los objetivos y responde adecuadamente a la interacción con el usuario.

6. Resultado
   En este punto se ve el resultado obtenido de todo el proceso anterior y cuál será la apariencia final de la aplicación.

7. Presupuesto
   En este punto se calcula el coste de los recursos empleados para realizar el proyecto.

8. Conclusión
   En base a los resultados obtenidos y todo el desarrollo del proyecto se hace una pequeña reflexión final viendo también las posibles líneas futuras del mismo.

9. Referencias
   Se verán todas las fuentes consultadas para la realización del mismo.

2. Estado del arte

Podríamos definir un dispositivo móvil como: "aparato de pequeño tamaño, con algunas capacidades de procesamiento, móviles o no, con conexión permanente o intermitente a una red, con memoria limitada, diseñados específicamente para una función pero que pueden llevar a cabo otras funciones más generales". http://es.wikipedia.org/wiki/Dispositivo_m%C3%B3vil

El primer teléfono móvil del mercado, fue desarrollado por Motorola a principios de los 80, fue el DynaTac 8000x cuya única funcionalidad era poder realizar llamadas. Con los años y una increíble evolución de las tecnologías se fueron consiguiendo funciones complementarias como enviar mensajes de texto, reproducir contenido multimedia, conexión a internet, cámara, GPS, pantallas táctiles, etc. poco a poco estos dispositivos iban brindando más posibilidades hasta llegar a lo que hoy conocemos como "smartphones" que se asemejan a pequeños ordenadores de bolsillo. Su origen data del año 1993. Aquellos primeros modelos se usaban como teléfonos de empresa y sus precios eran prohibitivos para la mayoría de los consumidores aparte que no alcanzaban ni una mínima parte de las funcionalidades que poseen hoy en día.

Estos teléfonos inteligentes cuentan con sistemas operativos muy avanzados que permiten la integración de software ofreciendo las mismas funcionalidades que un ordenador pero a un nivel más bajo de rendimiento. Estos sistemas operativos soportan una gran variedad de aplicaciones permitiendo una gran personalización del dispositivo aumentando sus funcionalidades hasta sorprendentes resultados.

Se puede encontrar una gran variedad de dispositivos en el mercado, como pueden ser PDAs, teléfonos móviles, portátiles, lectores de libros electrónicos, tabletas digitales, etc.

Su capacidad de darnos acceso a todo tipo de información en cualquier momento y desde cualquier lugar les ha vuelto indispensables en nuestro día a día. Contar con conexión a internet, pudiendo consultar nuestro correo electrónico, ver las noticias, su función de GPS, el acceso a cualquier tipo de contenido que necesitemos en un momento dado, etc. De esta manera el uso de ordenadores, televisiones y otros dispositivos están pasando a un segundo plano.

Si se quiere desarrollar una aplicación para un dispositivo móvil es importante conocer las plataformas de desarrollo existentes en el mercado. Actualmente las plataformas más populares en los terminales móviles son Symbian, IOS, Android, RIM y Windows Phone, aunque existen otras pero mucho menos utilizadas. A continuación se describirá cada una de ellas viendo sus características, sus fortalezas y debilidades, etc. para poder decidir cual será la mejor opción para las necesidades que deseamos cubrir.

Se puede encontrar información acerca de la evolución de los dispositivos móviles en: http://www.cheapdslmodem.org/mobile-phone.html/.

2.1 Sistemas operativos para móviles

En la Actualidad los SO móviles predominantes son Symbian, IOS, Android, RIM y Windows Phone.

Figura 1. Dispositivos móviles con los SO más extendidos en la actualidad, de izq a dcha: Symbian, Android, iOS, RIM y Windows Phone

2.1.1 Android

2.1.1.1 Información general

Android es el SO basado en Linux desarrollado por Google, sigue una filosofía de software libre y código abierto. Está desarrollado en Java, permite ejecutar procesos en segundo plano, soporta gráficos en 2D y 3D, ofrece una buena interfaz de usuario y nos proporciona una base de datos SQLite embebida.

El código de Android está liberado bajo licencia Apache v2.0 (a excepción de las modificaciones del kernel que han sido liberadas bajo la GNU GPLv2)

En 2005 Android empezó a dar sus primeros pasos, un sistema operativo para móviles diseñado para competir con: Apple (iPhone), Microsoft (Windows Mobile), Nokia (Symbian), Palm (WebOs), Research In Motion (Blackberry) y Samsung (Bada).

El primer terminal en salir al mercado con Android fue el HTC G1 (Dream en España) lanzado por T-Mobile en septiembre de 2008.

Android también ha hecho incursiones en *tablets* de la mano de compañías como HP, Dell, Notion INC o MSI (por citar algunos), eBooks (como el Alex de Grammata o el Nook de Barnes&Noble) e incluso la anunciada Google TV.
Algunas de sus características son las siguientes:

- Kernel basado en Linux.
- Framework de aplicaciones que permite reutilizar y remplazar sus componentes.
- Navegador web integrado basado en Webkit.
- Gráficos optimizados 2D (librería propia) y 3D (basados en OpenGL ES).
- SQLite para almacenamiento de datos.
- Soporte multimedia para los formatos más utilizados de sonido, vídeo e imágen (MPG4, H.264, MP3, AAC, AMR, JPG, PNG, GIF).
- Soporte para telefonía GSM.
- Soporte Bluetooth.
- Soporte EDGE.
- Soporte 3G.
- Soporte Wifi.
- Soporte para cámara.
- Soporte GPS.
- Soporte compás.
- Soporte acelerómetro.
- Gran entorno de desarrollo que incluye: documentación, emulador de dispositivos, herramientas de debug y análisis de uso de memoria/CPU, plugin para el entorno de desarrollo Eclipse y varias utilidades complementarias.

2.1.1.2 Arquitectura

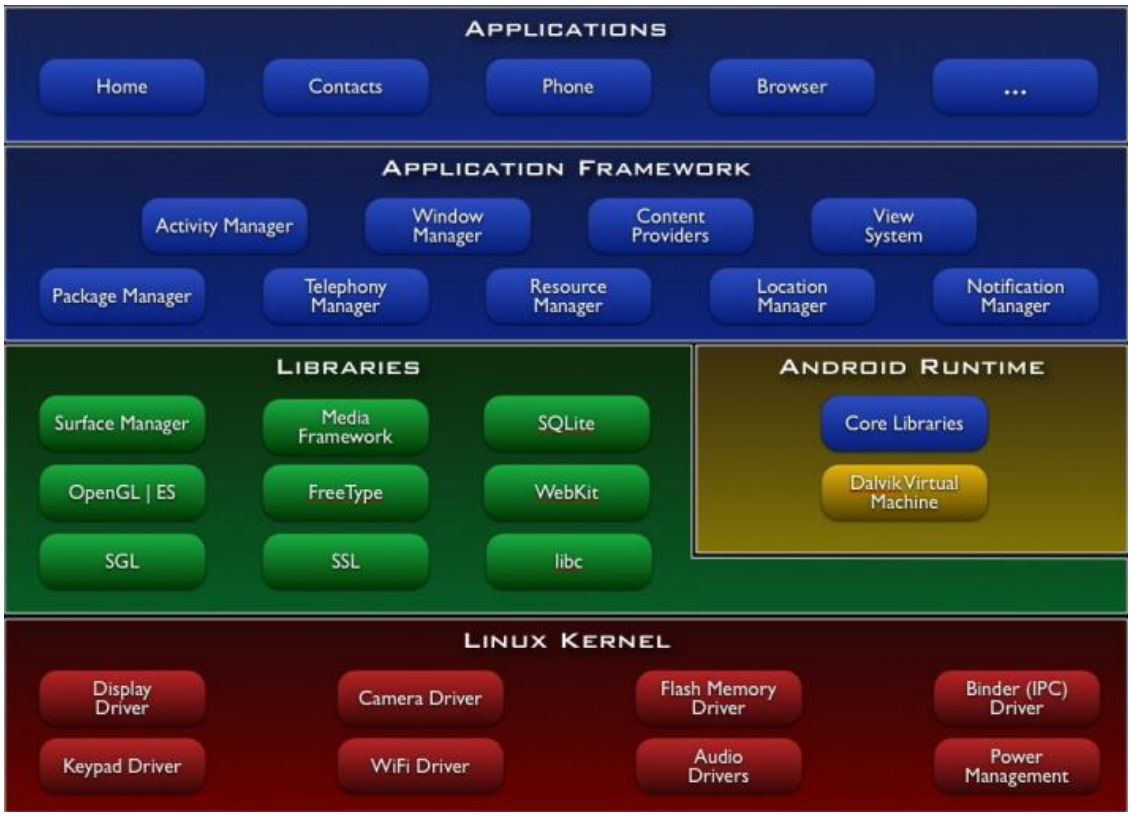


Figura 2. Ilustración de la arquitectura de Android

Vemos las diferentes capas de esta arquitectura:

Kernel de Linux

Es la capa mas baja del sistema, se encarga de interactuar con el hardware y contiene para ello todos los drivers necesarios. Para cada elemento de hardware del teléfono tendremos disponible un driver dentro del kernel que permite utilizarlo desde el software. Android está basado en el kernel de Linux versión 2.6, similar al que puede incluir cualquier distribución de Linux, como Ubuntu, solo que adaptado a las características de los dispositivos móviles. El desarrollador llega a esta capa a través de las librerías disponibles en capas superiores, haciéndose así tranparentes para este las características precisas de cada teléfono. El kernel también se encarga de gestionar los diferentes recursos del teléfono (energía, memoria, etc.) y del sistema operativo (procesos, elementos de comunicación, etc.).

Librerías

Se encuentran por encima del Kernel y suelen estar programadas en C/C++ y compiladas para la arquitectura hardware específica del teléfono, no se suelen acceder directamente ya que se usan las de la capa superior (framework). Habitualmente las hace el fabricante y las instala en el dispositivo antes de ponerlo a la venta. Las librerías dan funcionalidad a las aplicaciones para tareas que se repiten con frecuencia, evitando tener que codificarlas cada vez y garantizando que se llevan a cabo de forma eficiente. Algunas librerías son: OpenGL, Webkit, SSL, SQLite, etc.

Maquina virtual (Dalvik)

Google definió su propia máquina virtual con su propio bytecode. Esta basada en registros y no en la pila, optimizada para el uso en dispositivos móviles con sus restricciones de hardware y la necesidad de ahorro de energía. Dado que las aplicaciones están escritas en Java primero se llama a un compilador común de Java y posteriormente se convierte el bytecode de Java a bytecode de Dalvik. Los ejecutables que se generan con el SDK de Android tienen la extensión .dex. Tras la compilación las aplicaciones estarán listas para distribuirse con la total garantía de que podrán ejecutarse en cualquier dispositivo Android que disponga de la versión mínima del sistema operativo que requiera la aplicación. Su diseño hace posible que se ejecuten varias máquinas virtuales simultáneamente, ejecutándose cada aplicación en su propia máquina virtual y por tanto dando mayor seguridad.

Framework de aplicaciones

Facilita un marco de desarrollo que estandariza y facilita la programación de aplicaciones Android. Formado por todas las clases y servicios que utilizan directamente las aplicaciones para realizar sus funciones. La mayoría de los componentes de esta capa son librerías Java documentadas mediante JavaDoc que acceden a los recursos de las capas anteriores a través de la máquina virtual Dalvik. El desarrollador tiene completo acceso a todos los APIs utilizados en las aplicaciones base. Esta diseñado para reutilizar componentes facilitando mucho el acceso a todos los componentes del sistema.

Aplicaciones

En esta última capa es donde tendremos todas las aplicaciones que tiene el dispositivo. Hay algunas preinstaladas como agenda, calendario, gestor SMS, navegador, etc. y por otro lado todas las instaladas por el propio usuario. Para descargar aplicaciones vendrá preinstalado también el Android Market(https://play.google.com/store). Esta capa incluye también los widgets que podemos situar en los diferentes escritorios. Debido al diseño abierto tenemos una flexibilidad total sobre el dispositivo pudiendo incluso sustituir las aplicaciones base por otras con la misma funcionalidad. Por tanto se puede tener un dispositivo personalizado a la medida del usuario.

Se puede consultar la arquitectura más detallada en:
http://os.ibds.kit.edu/downloads/sa_2010_braehler-stefan_android-architecture.pdf.

2.1.1.3 Versiones

Versión	Fecha	API Level	Descripción
1.0	Septiembre 2008		Primera versión, muy limitada.
1.1	Febrero de 2009	2	Estabilizaba la versión anterior y la hacía funcional. Versión del primer terminal: HTC Dream.
1.5 (Cupcake)	Mayo de 2009	3	Rediseño y mejora del interfaz de usuario, mejoras de rendimiento, nuevas funcionalidades.
1.6 (Donut)	Octubre de 2009	4	Nuevas características de usuario (nueva caja de búsqueda, mejorada la cámara y la galería, soporte VPN, nuevos indicadores de batería, mejoras de accesibilidad), mejoras en el Android Market, nuevas tecnologías

			(framework de búsqueda mejorado, reconocimiento de voz, soporte para gestos, nuevas resoluciones de pantalla...). Aun así navegación limitada, apps aún de poca importancia y personalización muy limitada.
2.0/2.1 (Eclair)	Enero de 2010	7	Android va tomando forma. Nuevas características de usuario (mejoras en los contactos, email, mensajería, cámara, teclado virtual, calendario), nuevas tecnologías (bluetooth 2.1, nuevas APIs en el framework). Añade video HTML 5, navegación mejorada, cuentas personalizadas de google con sus diferentes apps, etc.
2.2 (Froyo)	junio de 2010	8	Se produce un salto de calidad en usabilidad. Se añaden los atajos para apps, , nuevas características de usuario (nueva home, soporte Exchange mejorado, cámara mejorada, hotspot, teclado virtual multi-idioma), mejoras de rendimiento, nuevas tecnologías de plataforma (media framework, bluetooth), nuevos servicios para desarrolladores (Cloud to Device Messaging, nuevos informes de error), nuevas APIs para programadores (almacenamiento externo de apps, media framework, cámara, gráficos, backup de datos, política de seguridad de dispositivo, ui framework).
2.3 (Gingerbread)	noviembre de 2010	9	Similar a la anterior con mejoras en la interfaz y en la usabilidad a la hora de la personalización.
3.0/3.1/3.2 (Honeycomb)	febrero de 2011	11/12/13	Configuración más accesible de la cámara, rediseñó el teclado, mejoras en la interfaz, introdujo la app de chat e incorporó el procesador multi-core.
4.0 (Ice Cream Sanwich)	octubre de 2011	14/15	La nueva interfaz elimina los botones físicos de la pantalla, incluyendo 3 botones digitales, incluye widgets que permitirá que las coloquemos donde deseemos, presenta el nuevo sistema de desbloqueo facial, se puede acceder directamente a cualquier app, el administrador de tareas también se reforma, incorporando la opción de voz a texto. La cámara se prefecciona en cuanto a configuraciones y efectos).
4.1 (Jelly Bean)	Julio 2012	16	Widgets personalizables son más cómodos de organizar y se colocaran de forma más intuitiva. El teclado virtual cuenta con un sistema de texto predictivo mejorado y se ha cambiado la tipografía del sistema. Mejoras también en la cámara y en la barra de notificaciones haciéndola mas visual. Destaca Google Now que ofrece información personalizada para el usuario basándose en el historial web, los datos de localización y otros as-

	pectos. Otra novedad es Project Butter que optimiza el rendimiento. También cuenta con una mejora en la potencia del procesador que además está ideada para ahorrar energía.

Tabla 1. Versiones Android

Podemos consultar la página oficial de Android en: http://www.android.com/ así como su portal de desarrolladores: http://developer.android.com/index.html y el de España: http://www.android-spa.com/.

2.1.2 iOS

2.1.2.1 Información general

Sistema operativo de Apple para sus dispositivos IPhone, iPod Touch, e iPad. Apple no permite la instalación de iOS en hardware de terceros. iOS se deriva de Mac OS X, que a su vez está basado en Darwin BSD, y por lo tanto es un sistema operativo Unix. El 9 de enero de 2007 fue presentado y el 29 de junio fue cuando salió al mercado en EEUU. El 6 de marzo de 2008 salió un Kit de Desarrollo para el dispositivo. Para todo tipo de movimientos se necesita iTunes ya sea a la hora de actualizar el SO como sincronización de archivos, creación de aplicaciones propias, etc. Posteriormente el 11 de julio apareció la App Store(http://store.apple.com) oficial de Apple que supuso una revolución para las aplicaciones móviles

Con todo esto el crecimiento fue muy rápido siendo una de las mayores tiendas de aplicaciones móviles. iOs siguió evolucionando suponiendo un gran avance la aparición del iPhone 3G con un diseño y potencia que le daban una gran experiencia de usuario. La interfaz de usuario de iOS está basada en el concepto de manipulación directa, usando gestos multitáctiles. Los elementos de control consisten de deslizadores, interruptores y botones. La respuesta a las órdenes del usuario es inmediata y provee de una interfaz fluida.

2.1.2.2 Arquitectura

Es una arquitectura basada en capas, a la hora de desarrollar se trabaja con las capas de alto nivel que son las que contienen los servicios y tecnologías necesarias. Las capas de bajo nivel controlan los servicios básicos.

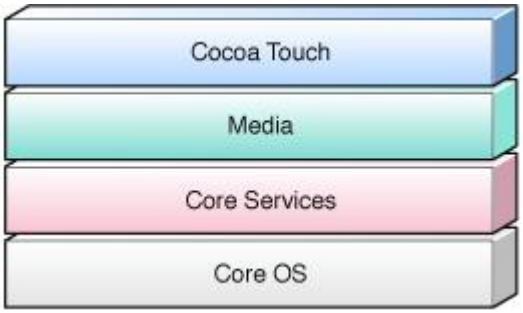


Figura 3. Arquitectura iOS

Cocoa Touch

Esta capa provee de la infraestructura necesaria para el desarrollo de aplicaciones iOS. Posee un conjunto de Frameworks que proporciona el API de Cocoa, implementan las tareas de bajo nivel y solo es necesario invocarlas. Esta basada en Objective C

Esta capa está formada por dos Frameworks que es fundamental conocer bien para desarrollar en iOS:

- Framework Foundation: define las clases básicas, un API usado por cualquier tipo de programa Cocoa.
- UIKit: provee de todas las clases que una aplicación necesita para construir y gestionar su interfaz de usuario.

Media

Esta capa provee a la capa superior de la tecnología necesaria para soportar 2D y 3D, audio y vídeo. Basada en C y Objective C.

Core Services

Contiene los servicios fundamentales del sistema que usan todas las aplicaciones.

Core OS

Contiene las características de bajo nivel: ficheros del sistema, manejo de memoria, seguridad, drivers del dispositivo, etc.

Para más detalles sobre la arquitectura de iOS:
http://www.techotopia.com/index.php/IPhone_iOS_5_Architecture_and_SDK_Frameworks.

2.1.2.3 Versiones

Versión	Fecha	Características
iOS 1.x(de la 1.0 a la 1.1.5)	Junio 2007	De la versión 1.0 a la 1.1.5. Incorporaba aplicaciones básicas como Mail, Fotos, Calculadora, etc.
iOS 2.x(de la 2.0 a la 2.2.1)	Julio 2008	Instalada de fábrica con el iPhone 3G. El fundamental cambio es la aparición del App Store.
iOS 3.x(de la 3.0 a la 3.2.2)	Junio 2009	Disponible originalmente a través del iPhone 3GS. Lo mas destacable son las notificaciones Push. Liberación de un SDK con más de 1000 APIs que los desarrolladores podían aprovechar en sus propios proyectos.
iOS 4.x(de la 4.0 a la 4.3)	Junio 2010	Llega con la aparición del iPhone 4. Muchos cambios, contando únicamente con todas sus funcionalidades el iPhone 3GS y el 4G. Multitarea, carpetas,
iOS 5.x	Junio 2011	Lanzada con el iPhone 4S. Revivió algunos dispositivos antiguos como el Ipod Touch 2G y el Iphone 3G. Como novedades el centro de notificaciones y la integración de iCloud
iOS 6.x	Otoño 2012	Siri en más lenguajes y con grandes mejoras. Mapas 3D Flyover sustituyen a Google Maps. Quizás venga de la mano de iPhone 5

Tabla 2. Versiones iOS

Los últimos SO de iOS dejaban de ser compatibles con algunos dispositivos, en el caso del próximo iOS 6 será compatible con:

- iPhone 3GS
- iPhone 4
- iPhone 4S
- iPad 2
- Nuevo iPad (iPad 3)
- iPod Touch 4G
- Apple TV 2G
- Apple TV 3G

No compatible con:

- iPad Primera Generación
- iPod Touch 3G

Las versiones anteriores ya quedaron demasiado obsoletas para mantener estos nuevos sistemas operativos. Pero hay que tener cuenta que aunque actualicemos a la última ver-

sión para muchos dispositivos hay funciones que se pierden como Siri, lectura offline, etc.

Se puede consultar la web de Apple en:http://www.apple.com y su principal portal de desarrolladores en: http://developer.apple.com.

2.1.3 Otros

2.1.3.1 Symbian

Symbian OS fue producto de la alianza de varias empresas de telefonía móvil, entre las que se encuentran Nokia, Sony Ericsson, PSION, Samsung, Siemens, Arima, Benq, Fujitsu, Lenovo, LG, Motorola, Mitsubishi Electric, Panasonic, Sharp, etc.

En 2008, Nokia decidió comprar Symbian con la idea de establecer la Fundación Symbian y convertir este sistema operativo en una plataforma abierta, pero lo acaba haciendo bajo la licencia Nokia Symbian, poniendo el desarrollo en manos principalmende de Accenture. Symbian tendrá soporte hasta el año 2016, dejando a día de hoy ya de presentar terminales. En Nokia 803(sucesor del Nokia N8) podría ser el último con este SO. Symbian tuvo una época dorada en la que lideró el mercado para dar paso a una nueva etapa con Windows Phone como protagonista.

Symbian tiene su matriz más lejana en el EPOC, un sistema operativo gráfico creado en los años 80 por la compañía Psion, utilizado en PDA's y Handhelds. EPOC-R5 es un sistema operativo de 32 bits multihilo optimizado para dispositivos móviles que utilizan comunicaciones inalámbricas. SO de escritorio multitarea (multi tasking), multilectura y con protección de memoria.

El lenguaje habitual para programación en Symbian es C++, pero se han ido desarrollando herramientas para programar en Java, Python, OPL, etc. por lo que hay un gran abanico de posibilidades a la hora de desarrollar para Symbian OS.

La última actualización del sistema operativo es Symbian Belle FP1 (14 de abril de 2012), surge con grandes mejoras y un cambio importante en la interfaz gráfica.

http://symbian.nokia.com

2.1.3.2 Blackberry OS

RIM (Research In Motion) es la compañía canadiense fabricante y promotora de Blackberry, desarrolla su propio software para sus dispositivos usando C++, C y la tecnología Java.

Blackberry OS es el sistema operativo creado por RIM, es un sistema multitarea enfocado principalmente a un uso profesional con sus servicios asociados de mensajería y gestión de correo dando una gran garantía de seguridad. Es un sistema de código cerrado con licencia propietaria.

La aparición del primer dispositivo con este SO se produce en 1999 con los terminales Handheld que permitían acceder a nuestras cuentas de correo electrónico, navegación web, y conexión a programas de gestión de correo y agenda como Microsoft Exchange o Lotus Notes, además de ofrecernos los servicios y características propias de un teléfono móvil.

En Junio de 2012 aparece la última versión estable, la 7.1, compatible con Blackberry Bold 9900, Torch 9860, Curve 9360/9380/9220.

http://es.blackberry.com/

2.1.3.3 Windows Phone

Windows Phone 7 aparece en Octubre de 2010 como un sistema operativo móvil desarrollado por Microsoft(modelo propietario), como sucesor de la plataforma Windows Mobile. Este no será compatible con Windows Mobile 6 y estará menos orientada al mercado empresarial. Con Windows Phone 7 Microsoft ofrece una nueva interfaz de usuario e integra varios servicios en el sistema operativo.

Se basa en el núcleo del sistema operativo Windows CE y cuenta con un conjunto de aplicaciones básicas utilizando las API de Microsoft Windows y estéticamente está diseñado para ser similar a las versiones de escritorio de Windows.

El 20 de Junio de 2012 presentaron su nuevo SO Windows Phone 8 cuyo lanzamiento se prevé para finales de año. Con esta nueva versión pretenden competir con iOS y Android. Esta actualización supondrá el comienzo de la fragmentación ya que los dispositivos actuales no podrán actualizarse. El lenguaje empleado para hacer desarrollos para este SO es C#.

http://www.microsoft.com/windowsphone/

2.1. 4 Problema de la fragmentación.

Esto se refiere a la coexistencia de diferentes versiones del SO sobre muy diferentes terminales móviles lo que supone un completo ecosistema de sistemas con sus problemas de compatibilidades entre ellos y sus necesidades de actualización a versiones superiores del sistema operativo.

Este problema se produce sobretodo en Android ya que cuenta con una gran cantidad de fabricantes distintos y un gran número de actualizaciones del sistema operativo. Esto supone problemas de estandarización, compatibilidad y pone muchos inconvenientes a la hora de que un usuario pueda tener su móvil correctamente actualizado. A su vez también complica la labor del desarrollador a la hora de conseguir que las aplicaciones funcionen adecuadamente para la mayor parte de ellos, bien por temas de las versiones así como por ejemplo la gran variedad de tamaños de pantalla diferentes.

En el siguiente gráfico podemos ver la distribución de las versiones Android en el mercado (http://developer.android.com/about/dashboards/index.html):

Version	Codename	API Level	Distribution
1.5	Cupcake	3	0.2%
1.6	Donut	4	0.5%
2.1	Eclair	7	4.7%
2.2	Froyo	8	17.3%
2.3 - 2.3.2	Gingerbread	9	0.4%
2.3.3 - 2.3.7		10	63.6%
3.1	Honeycomb	12	0.5%
3.2		13	1.9%
4.0 - 4.0.2	Ice Cream Sandwich	14	0.2%
4.0.3 - 4.0.4		15	10.7%

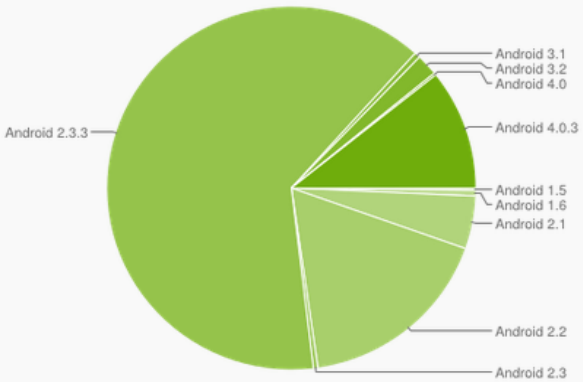


Figura 4. Cuota de mercado de las versiones de Android

Podemos observar una total dispersión donde lejos de tener la mayor cuota la última versión (Ice Cream Sandwich) esta solo cuenta con un 10,9% a 10 meses de su lanzamiento. Las versiones Honeycomb destinadas a su uso concreto en Tablets tampoco contaron con demasiado éxito pudiendo ver como la versión 2.3 Gingerbread es la más extendida con un 64%.

Para hacernos una idea del fenómeno de la fragmentación en función de los diferentes dispositivos podemos observar la siguiente imagen (http://www.bgr.com/2012/05/16/android-fragmentation-visualized-opensignalmaps/):

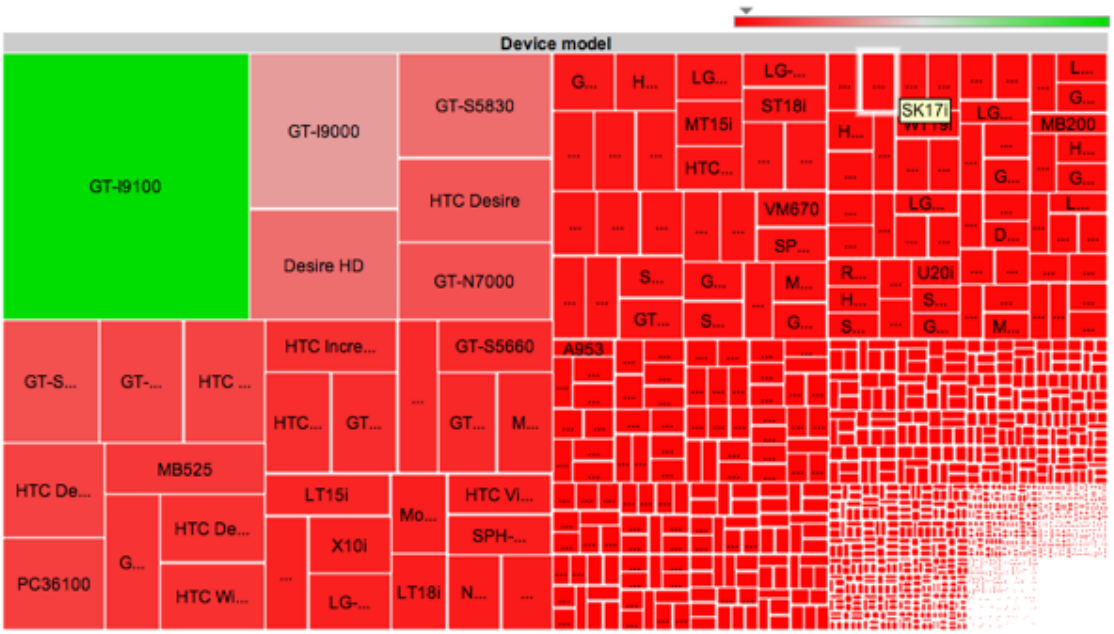


Figura 5. Ilustración sobre la fragmentación en Android

Se pueden encontrar cerca de 4000 dispositivos Android distintos. La marca predominante de todas las que llevan instalado Android es Samsung (con un 40%) siendo sus modelos más extendidos el Samsung Galaxy S y el Samsung Galaxy S II (10% de la cuota de mercado).

En iOS este problema se reduce mucho pero aun así existe, ya que podemos actualizar nuestro dispositivo a la última versión del SO de una manera muy sencilla a través del iTunes pero podremos estar perdiendo parte de la funcionalidad del mismo ya que parte de la misma solo estarán disponibles para determinados dispositivos. Además va dejando de dar soporte a las primeras versiones para evitar este tipo de problema en la mayor medida posible.

2.1.5 Conclusión

En la siguiente página: http://www.idc.com/getdoc.jsp?containerId=prUS23503312 podemos encontrar una comparativa de las cuoteas de mercado de los diferentes sistemas operativos móviles:

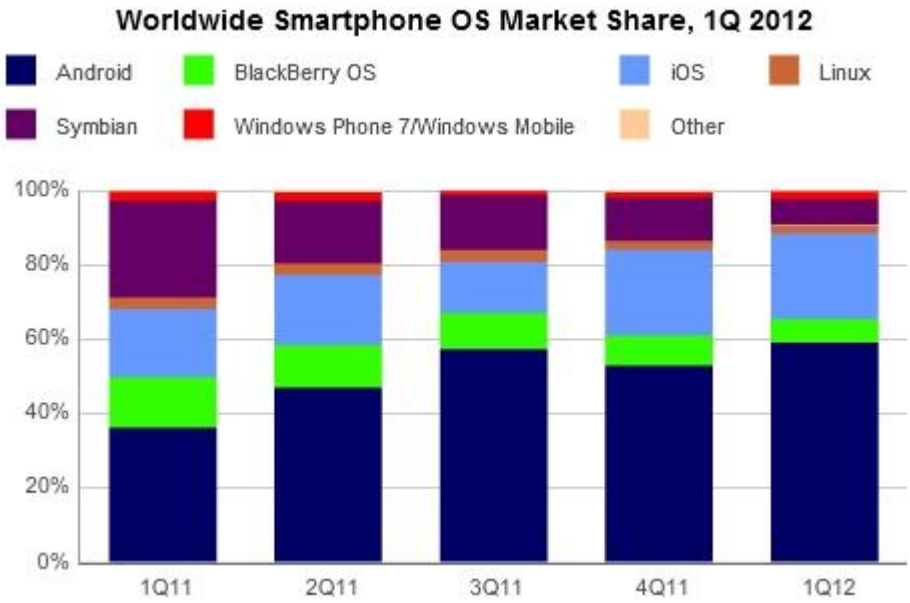


Figura 6. Comparativa de la evolución de los SO móviles entre 2011 y 2012

Como se puede observar, de las cinco plataformas, Symbian, RIM y Windows Mobile han perdido cuota de mercado a lo largo de estos años. La más afectada ha sido Symbian al perder un total de 20% de cuota en un año, siendo una cifra bastante significativa. RIM pierde algo más del 7%. Y Windows Phone un 0.4%. Este terreno cedido por las tres plataformas ha sido tomado por Android e IOS que han aumentado su proporción, especialmente Android, llegando a conseguir casi un 30% más desde el 2009.

En el primer trimestre de 2012 a nivel mundial los porcentajes de cuota de mercado fueron:

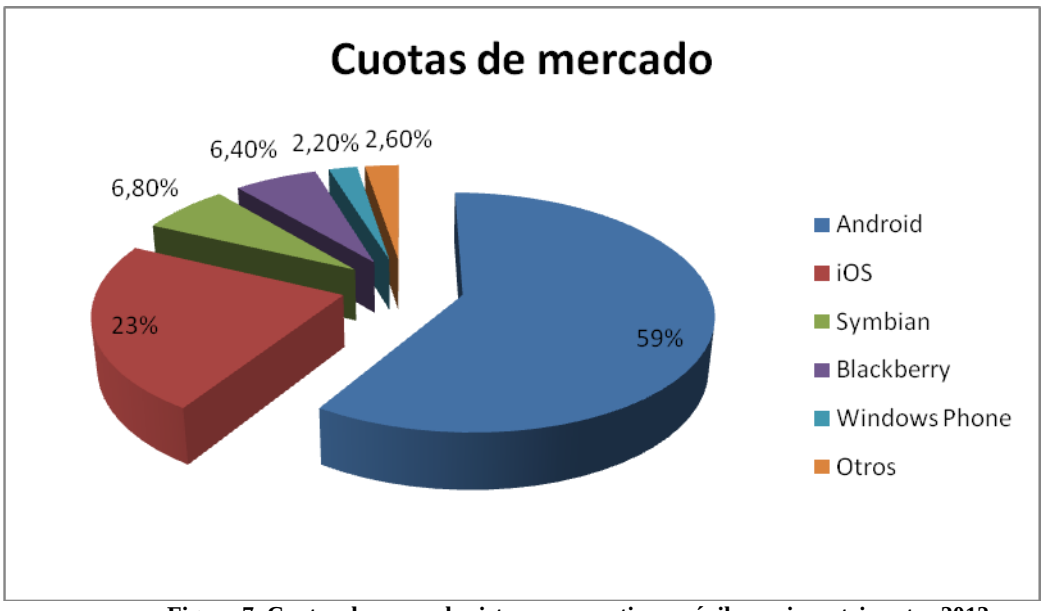


Figura 7. Cuotas de mercado sistemas operativos móviles, primer trimestre 2012

En un estudio tomado en el segundo cuatrimestre de 2010 analizando las diferentes tiendas para las distintas plataformas móviles. De todas ellas, la plataforma IOS para iPhone poseía el mayor número de aplicaciones con más de 200.000, seguida de Android superando las 70.000. Con estos datos se puede llegar a la conclusión que tanto Android como IOS iPhone poseen una mayor actividad de desarrollo comparada con el resto de plataformas.

Si tenemos en cuenta los datos correspondientes a las cuotas de mercado y la alta actividad en los desarrollos de aplicaciones, de entre todas las plataformas más populares tanto Android como IOS parecen ser las más indicadas para realizar desarrollos de aplicaciones. Ambas están en pleno auge comiendo terreno a las demás plataformas y llevando un buen ritmo de crecimiento. A su vez al tener ambas una alta actividad de desarrollo proporciona mayores recursos y mayor movimiento en las tiendas relativo a compras y ventas.

2.2 Wordpress

WordPress es una avanzada plataforma semántica de publicación personal orientada a la estética, los estándares web y la usabilidad. WordPress es libre y, al mismo tiempo, gratuito. Es un sistema de gestión de contenido enfocado a la creación de blogs, pero ha ido evolucionando haciendo posible crear cualquier tipo de sitio web Desarrollado en PHP y MySQL, bajo licencia GPL. Las causas de su enorme crecimiento son, entre otras, su licencia, su facilidad de uso y sus características como gestor de contenidos.

Su instalación es muy sencilla y rápida, es necesario para la misma tener instalado php y mysql. La última versión estable disponible para Wordpress es la 3.4.

Descargamos la última versión de Wordpress en:

http://wordpress.org/download/

Los pasos a seguir una vez descargado son los siguientes:

1. Descarga y extrae el paquete de WordPres.

2. Crea una base de datos para WordPress en el servidor web, así como un usuario de MySQL que tenga todos los privilegios para accederla y modificarla.

3. Renombra el archivo wp-config-sample.php con el nombre wp-config.php.

4. Se abre wp-config.php en un editor de texto y completa los datos para la base de datos.

5. Colocar los archivos de WordPress en la localización deseada en el servidor web:
   - Si se quiere integrar WordPress en la raíz del dominio (ej. http://example.com/) mueve o sube los contenidos del directorio donde WordPress fue extraído en el directorio raíz del servidor web.
   - Si quieres que la instalación tenga su propio subdirectorio en tu sitio web (ej. http://example.com/blog/) renombra el directorio wordpress al nombre que desees y muévelo o súbelo a tu servidor web. Como ejemplo, si quieres tu instalación en un subdirectorio llamado "blog" deberás renombrar el directorio "wordpress" a "blog" y subirlo al directorio raíz del servidor web.

6. Por último se ejecuta el script de instalación de WordPress accediendo wp-admin/install.php en tu navegador web preferido.
   - Si se instaló WordPress en el directorio raíz se debe ir a http://example.com/wp-admin/install.php.
   - Si se instaló WordPress en su propio subdirectorio llamado blog, se accede a http://example.com/blog/wp-admin/install.php

Si todo va bien saldrán unos pasos para definir el título de la página así como el email, a continuación se nos generará una contraseña y por último tendremos la siguiente pantalla para acceder a nuestro Wordpress:

Figura 8. Pantalla inicio Wordpress

Infraestructura

- WordPress, es un sistema de publicación web basado en entradas ordenadas por fecha, páginas estáticas, etc.
- La estructura y diseño visual del sitio depende del sistema de plantillas.
- La filosofía de WordPress apuesta decididamente por la elegancia, la sencillez y las recomendaciones del W3C pero depende siempre del tema a usar. "Classic", por ejemplo es el tema que viene por defecto y que es válido como XHTML Transicional y CSS.
- Separa el contenido y el diseño en XHTML y CSS, aunque depende del tema que se esté usando. No obstante, el código que se intenta generar en los posts, apuesta por esta característica forzando un correcto marcado.

- La gestión y ejecución corre a cargo del sistema de administración con los plugins y los widgets que usan los themes.

  En Wordpress tenemos varios tipos de contenido:

- Entradas (Posts) son el contenido por defecto que se utiliza para entradas de blog. Están destinadas a contenido dinámico, donde la primera entrada del blog se modifica al añadirse otra nueva.

- Páginas (Pages) son el tipo de contenido por defecto de WordPress, destinado a contenido estático. A diferencia de las entradas, estas no varían con el tiempo.

- Tipo de Contenido Personalizado, definidos por el usuario a su medida, incrementan la flexibilidad de WordPress.

Las Taxonomías son la forma de clasificar todo tipo de contenido en Worpress. Tenemos categorías que son taxonomía jerárquica y etiquetas que son taxonomía no jerárquica.

WordPress tiene la gran ventaja de ser utilizado tan extensamente en el mundo, que hay una gran cantidad de plugins desarrollados para el mismo aportando todo tipo de funcionalidades.

Un ejemplo de diseño con Wordpress sin niguna apariencia de blog y con gran atractivo visual es:

http://www.serj.ca/

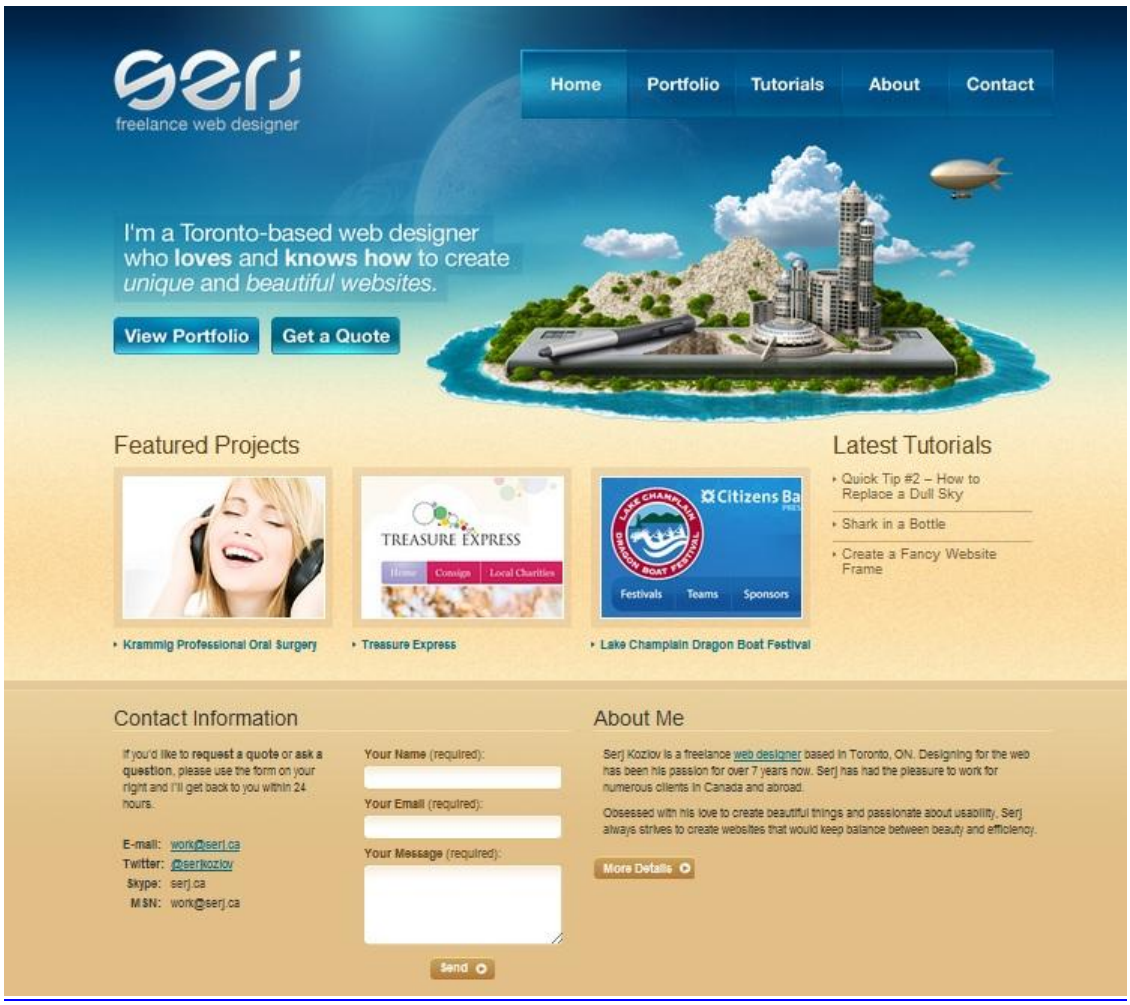


Figura 9. Ejemplo página Wordpress

También se han desarrollado varios plugins para la obtención del contenido de portales Wordpress. Esto hace posible desarrollar clientes que permitan acceder a la información incluso editarla desde otros dispositivos y aplicaciones.

2.3 Codeeta

Es un software que permite construir y exportar widgets con funcionalidad web avanzada de forma sencilla. Cuando se crea un widget con Codeeta, conecta automáticamente con los servicios web adecuados, crea la base de datos y ejecuta toda la lógica necesaria para conseguir funcionalidad web.

Permite:

- Creación de formularios web
- Recepción de pagos online
- Recepción de micropagos online
- Venta online de productos físicos y descargables
- Gestión de reservas online
- Gestión de contenidos

Codeeta dispone de una completa API que devuelve resultados en formato JSON.

Para más información: http://codeeta.com/

2.4 Formatos de comunicación

2.4.1 XML-RPC

Es un protocolo de llamada a procedimiento remoto que usa XML para codificar los datos. Funciona mediante intercambio de mensajes entre cliente y servidor, utilizando el protocolo HTTP para el transporte de los mensajes. XML–RPC utiliza peticiones POST de HTTP para enviar un mensaje, en formato XML

Tipos de datos:

- Array
- Base64
- Boolean
- Date/time
- Double
- Integer
- String
- Struct
- Null

Ejemplo de petición:

```
<?xml version="1.0"?>
<methodCall>
   <methodName>examples.getStateName</methodName>
   <params>
```

```
      <param>
         <value><i4>41</i4></value>
         </param>
      </params>
   </methodCall>
```

Ejemplo de respuesta:

```
<?xml version="1.0"?>
<methodResponse>
   <params>
      <param>
         <value><string>South Dakota</string></value>
         </param>
      </params>
   </methodResponse>
```

Se puede consultar la especificación completa para XML-RPC en: http://xmlrpc.scripting.com/spec.html

2.4.2 JSON

JSON (JavaScript Object Notation) es un formato ligero para el intercambio de datos. Es sencillo de leer y de escribir así como de interpretar por máquinas. Esta basado en un subconjunto de JavaScript. El formato de JSON es ampliamente reconocido por una gran variedad de lenguajes como Java, PHP, JavaScript, C++, C# y muchos otros. Esto hace que JSON sea un lenguaje de gran potencial para el intercambio de datos.

Esta constituido por dos estructuras soportadas en cualquier lenguaje de programación:

- Una colección de pares nombre / valor
- Una lista ordenada de valores

Elementos en JSON:

Un objeto se conforma de una llave de apertura, el nombre del objeto entrecomillado, dos puntos, el valor dado al objeto y una llave de cierra.

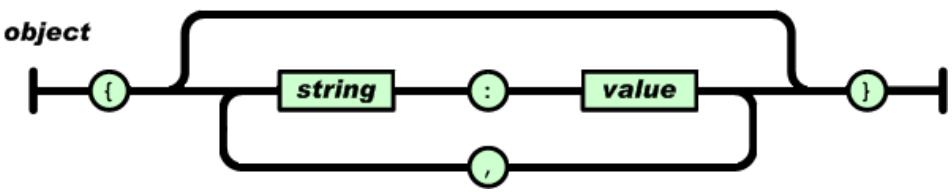


Figura 10. Objeto JSON

Ejemplo: { "objeto" : valor }

Un array es una colección de valores. Se define entre corchetes y los valores se separan por, (coma).

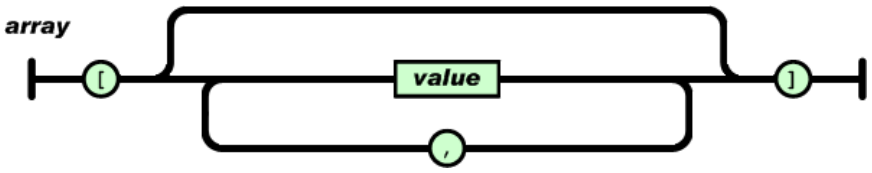


Figura 11. Array JSON

Ejemplo: "datos": [{"nombre1" : "valor1", "nombre2": valor2, "nombre3" : "valor3", "nombre4": "valor4"}]

El valor puede tomar cualquiera de los siguientes tipos de datos:

- String
- number
- object
- char
- array
- null
- boolean

Aquí se muestra un ejemplo de JSON frente al tradicional XML.

Si el fichero xml es:

view plainprint?

```
<?xml version="1.0" encoding="UTF-8" ?>
<poblaciones>
  <poblacion id="0">Alcobendas</poblacion>
  <poblacion id="1">Miraflores de la Sierra</poblacion>
  <poblacion id="2">San Fernando de Henares</poblacion>
</poblaciones>
```

Su documento JSON sería el siguiente:

view plainprint?

```
{"poblaciones":[
    {"poblacion": { "@id": "0", "#text": "Alcobendas" }}
,
    {"poblacion": { "@id": "1", "#text": "Miraflores de la Sierra" }}
,
    {"poblacion": { "@id": "2", "#text": "San Fernando de Henares" }}
]}
```

Página oficial de JSON: http://www.json.org/

2.5 Entornos de desarrollo

2.5.1 Eclipse

El Eclipse es un entorno de desarrollo integrado (IDE, Integrated Development Environment) que facilita enormemente las tareas de edición, compilación y ejecución de programas durante su fase de desarrollo. Aunque Eclipse soporta varios lenguajes de programación, como: Java, C, C++, PHP, VB, Perl, C#, Python, JavaScript, Delphi, COBOL, ADA, Ruby, etc.

Eclipse es una aplicación gratuita y de código abierto, disponible para su descarga en la página oficial:

http://www.eclipse.org/downloads/

No necesita instalación. Se descarga el programa, se descomprime y tal como está descomprimido, se empieza a usar.

Eclipse fue desarrollado originalmente por IBM como el sucesor de su familia de herramientas para VisualAge. Eclipse es ahora desarrollado por la Fundación Eclipse, una organización independiente sin ánimo de lucro que fomenta una comunidad de código abierto y un conjunto de productos complementarios, capacidades y servicios.

Eclipse proporciona el entorno de desarrollo solamente. Se considera uno de los mejores entornos de desarrollo para aplicaciones Java. En su caso es necesario disponer del Java Development Kit o JDK para poder compilar y ejecutar las aplicaciones desarrolladas.

Por tanto usaremos Eclipse y no otros entornos como NetBeans por la mayor madurez del plugin de android para Eclipse así como por lo visto anteriormente.

2.5.1.1 Plugin Android

ADT amplía las capacidades de Eclipse para que pueda configurar rápidamente nuevos proyectos de Android, se crea una interfaz de usuario de aplicación, se agregan los paquetes basados en la API de Android, se depuran las aplicaciones utilizando las herramientas del SDK de Android, e incluso permite exportar el aqrchivo apk con firma con el fin de distribuir la aplicación.

Descargar el Plugin ADT

Al iniciar Eclipse se selecciona **Ayuda → Instalar nuevo software** y en **Añadir** con nombre: "ADT Plugin" y URL: "https://dl-ssl.google.com/android/eclipse/ ".

En el cuadro de diálogo de software disponibles, se selecciona la casilla junto a Herramientas de Desarrollo y pulsamos **Siguiente**.

En la siguiente ventana, se muestran una lista de herramientas para ser descargadas. De nuevo hacemos indicamos **Siguiente**.

Lea y acepte los acuerdos de licencia y finalizamos.

Al finalizar la instalación, se debe reiniciar Eclipse.

Configurar el plugin ADT

Una vez instalado y reiniciado ADT Eclipse, se debe indicar la ubicación del Android SDK:

Se abre el panel de preferencias en **Ventana→Preferencias** y accediendo a **Android** en el panel izquierdo se accede al wizard donde se debe seleccionar el directorio del SDK.

Si no hay ningún error, ya está hecho el establecimiento de ADT y se puede continuar con el siguiente paso de la instalación del SDK.

Al instalar el plugin aparecerán dos nuevos iconos en la barra de herramientas, podemos verlos en la siguiente imagen debajo de la pestaña Navigate:

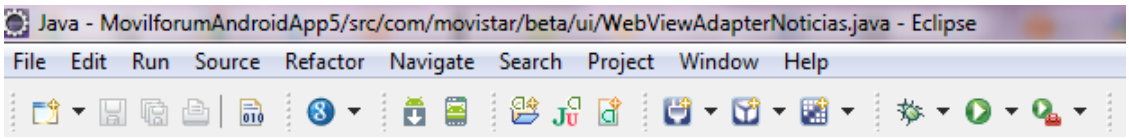


Figura 12. Barra de herramientas con iconos generados por plugin Android

Estas pestañas son el SDK Manager (a la izquierda) y el AVD (Android Virtual Devices).

SDK Manager

Para crear un dispositivo virtual de una versión en concreto de Android, es necesario primero haber instalado el software necesario para ejecutar dicha versión en el SDK. Este software es liberado por Google y se puede descargar mediante la aplicación SDK **Manager**. Se muestra a continuación su apariencia:

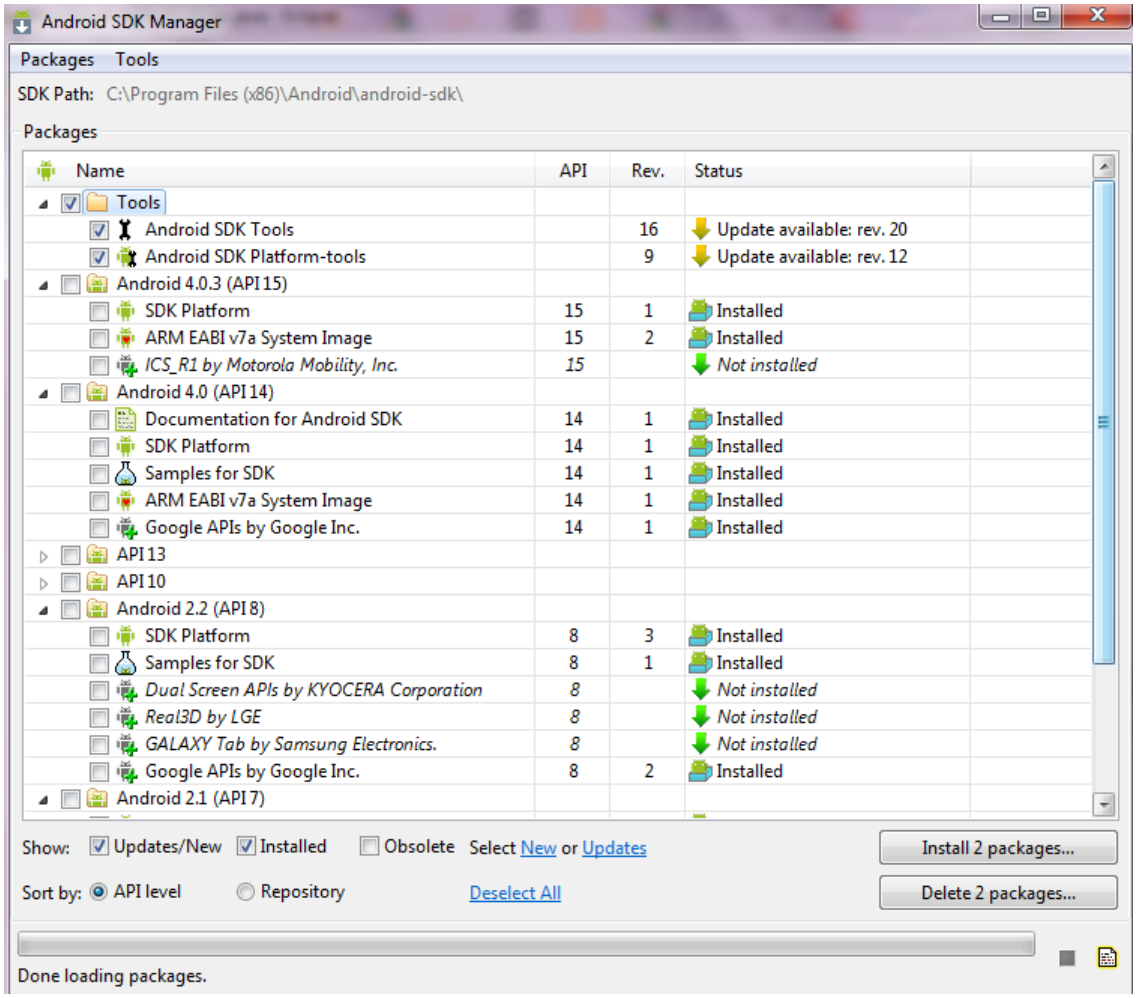


Figura 13. Vista SDK Manager

Esta vista indica las versiones disponibles de Android que no tenemos instaladas en el SDK. Para usar el emulador con una versión específica de Android, hay que instalar los paquetes de esa versión concreta. Una vez seleccionado y pulsado el botón instalar, el SDK descargará automáticamente los archivos necesarios y los instalará.

AVD (Android Virtual Devices)

Herramientas imprescindibles para los desarrolladores y testers, ya que permiten emular en una computadora los dispositivos móviles a los que apunta nuestra aplicación. Por defecto, cuando se instala el AVD Manager, no viene cargado ningún dispositivo virtual. Al pinchar sobre el botón de la barra de herramientas vemos lo siguiente:

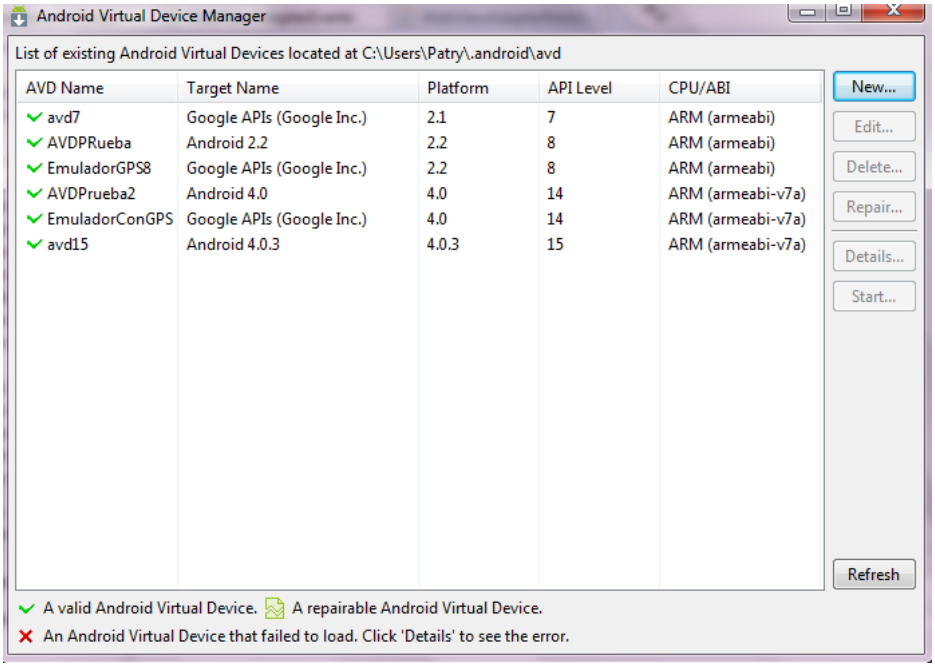


Figura 14. Vista Android Virtual Device

Para agregar un nuevo dispositivo pulsamos en **New** y aparecerá lo siguiente:

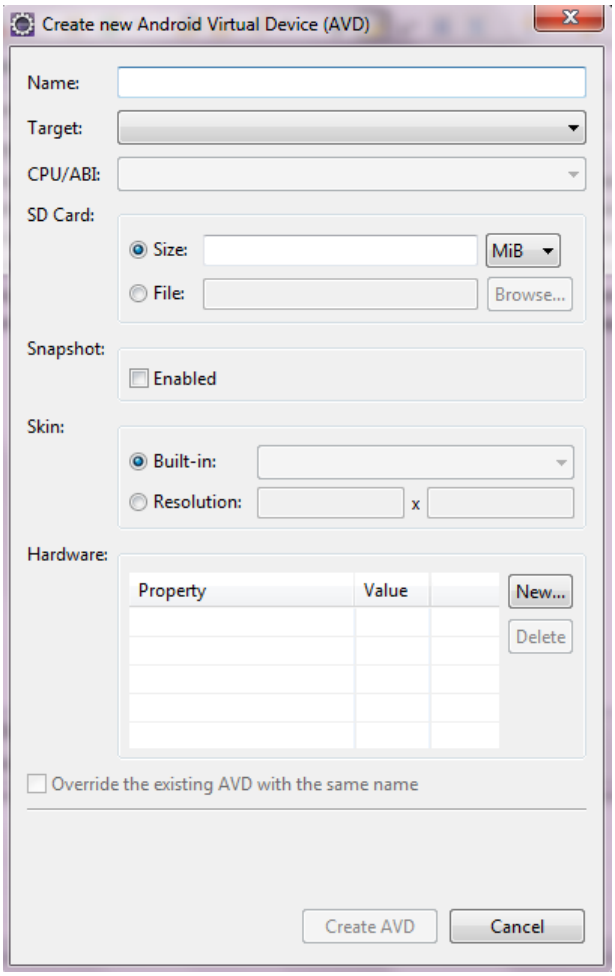


Figura 15. Creación nuevo AVD

Bastara con dar un nombre al nuevo dispositivo (name) e indicarle en que versión de Android queremos que corra (target).

Con esto tendremos el nuevo AVD bastará con seleccionarle en el momento de ejecutar nuestra aplicación. Para ejecutarla:

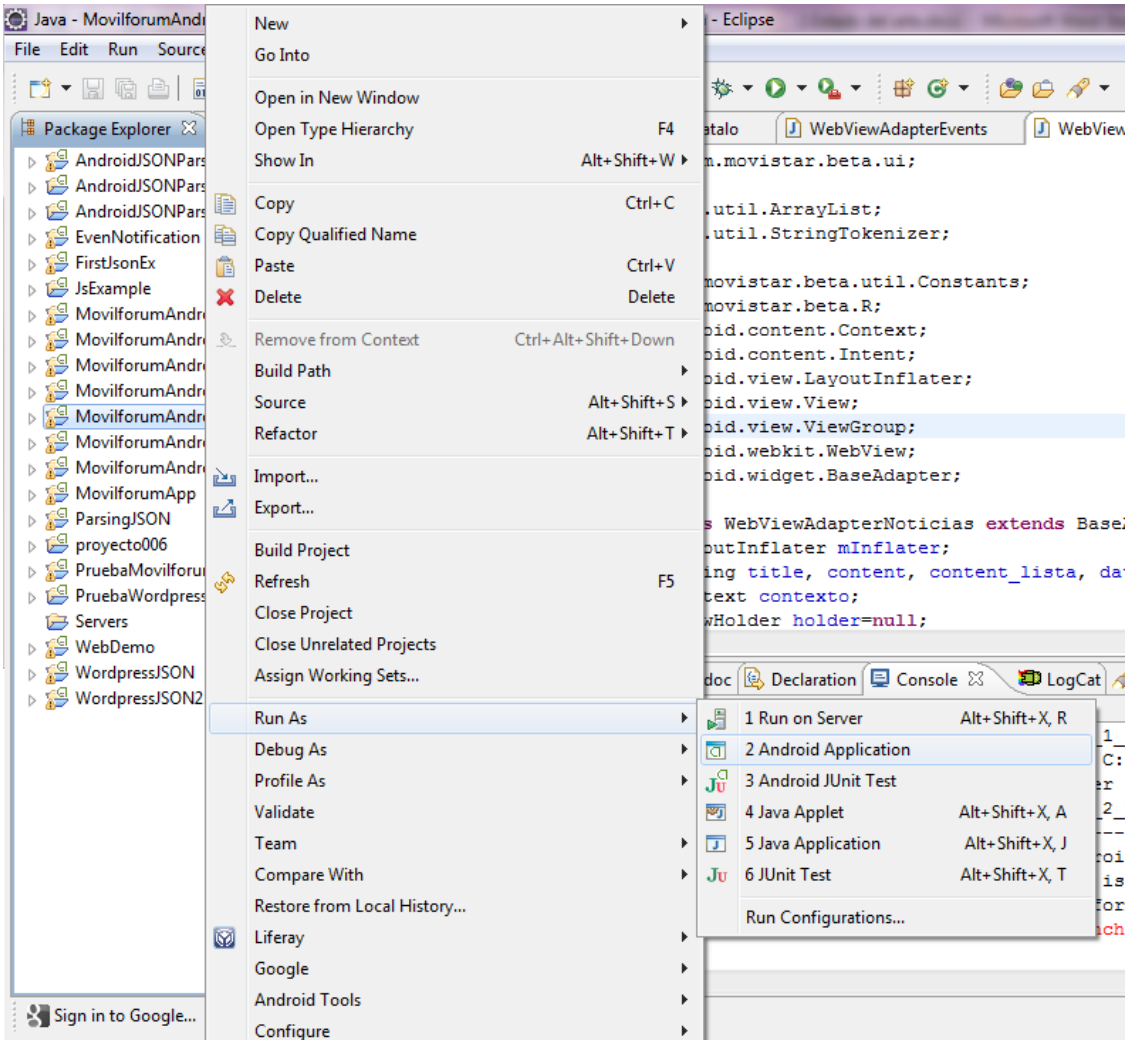


Figura 16. Vista ejecución aplicación

Y a continuación se nos permitirá seleccionar el dispositivo a emular:

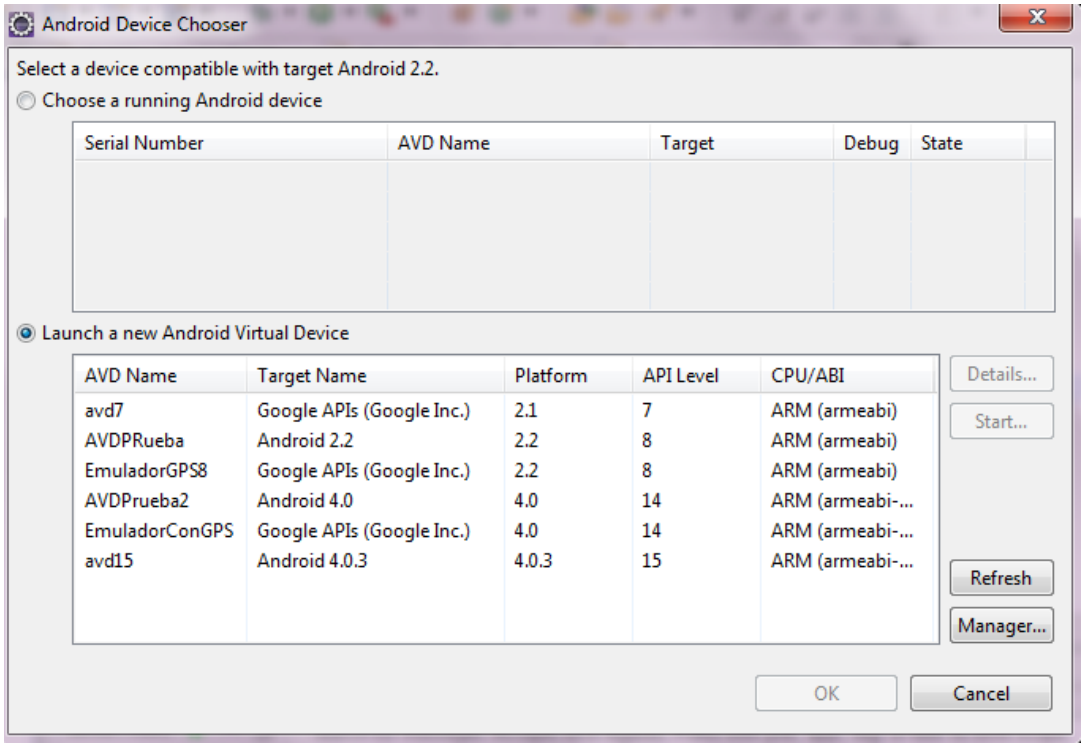


Figura 17. Vista selección AVD para ejecución

Abriéndose a continuación con una apariencia así:

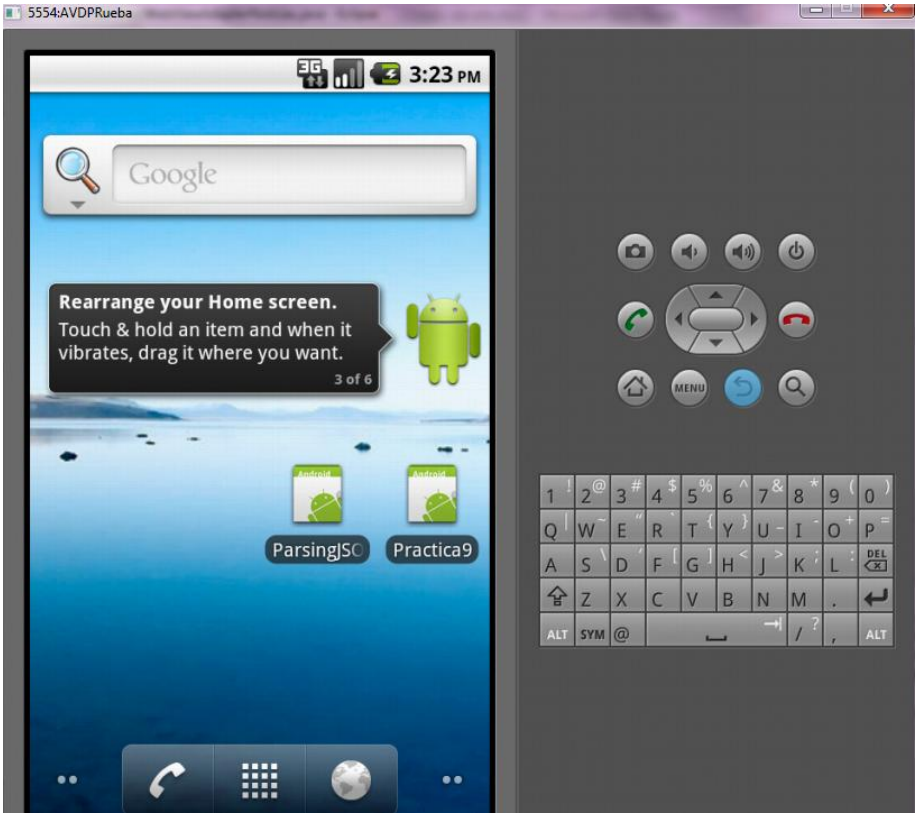


Figura 18. Vista emulador Android

2.5.2 Titanium

Appcelerator Titanium(http://www.appcelerator.com/) es un framework Javascript para desarrollar aplicaciones de escritorio y para móviles. En su versión para móviles, se puede utilizar para desarrollar aplicaciones para iPhone, Android y Blackberry. Se distingue de otros frameworks en que genera aplicaciones nativas en lugar de aplicaciones que se ejecutan dentro de un navegador.

Simplemente con tener conocimientos de HTML, JavaScript y CSS podremos crear nuestras propias aplicaciones pudiendo explotar gran parte de la funcionalidad del SO. Nos brinda por tanto la posibilidad de desarrollar aplicaciones sin tener conocimientos específicos de Objective C, Java o C++.

El IDE de Appcelerator para desarrollo es Titanium Studio, basado en Eclipse con el que crear los proyectos y editar los ficheros Javascript y el resto de recursos y lanzar los scripts de creación. Se hace uso de una librería que hace de puente entre la aplicación Javascript y los controles del sistema, por tanto todos los componentes visuales de la interfaz son nativos lo que supone un gran rendimiento y flexibilidad frente a otras herramientas similares.

Al empaquetar la aplicación, el JavaScript es transformado y compilado. Cuando se arranca la aplicación en el móvil, el código se ejecuta dentro de un engine Javascript, que será JavaScriptCore en IOS y Mozilla Rhino en Android/Blackberry. Existen librerías especificas para las Android e iOS por lo que hay que revisar cada desarrollo si se quiere que funcione en ambas.

A continuación se ve una captura del Titanium Studio cuya apariencia es muy similar a Eclipse:

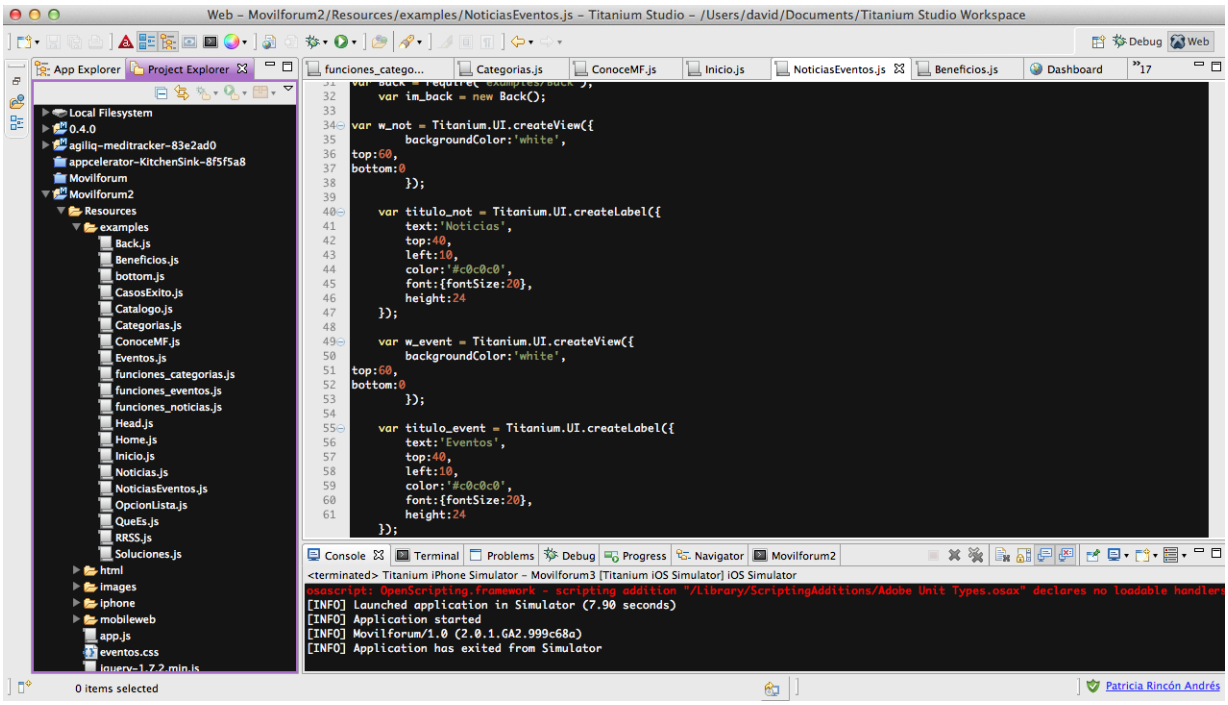


Figura 19. Vista Titanium Studio

Y al ejecutar la aplicación aparece el emulador correspondiente según el SO que simulemos en este caso iOS:

Figura 20. Vista emulador iOS en Titanium Studio

2.5.2.1 Comparativa con PhoneGap

Aunque no tengamos tantas posibilidades como desarrollando de forma directamente nativa con los lenguajes correspondientes se puede llegar a ganar mucho en tiempo desarrollo. Es más fácil encontrar programadores para estas tecnologías que para las propias de estos SO móviles.

Las aplicaciones que se generan son híbridas, lo que significa que no están por entero en el código nativo de la plataforma para la que se está desarrollando, pero tampoco son sólo aplicaciones web, pues tienen acceso a la interfaz de cada dispositivo.

La filosofía de este entorno de desarrollo es desarrollar el código una sola vez para ejecutarlo en todas partes. Hay otras herramientas para desarrollo en móviles con esta misma filosofía como Corona (lenguaje Lua) y Flex 4 y Adobe Air Mobile (ActionScript) pero dado que PhoneGap y Appcelerator son gratuitos y consideramos JavaScript una tecnología adecuada y conocida a gran escala nos centraremos en estos dos.

Vemos por tanto como funciona PhoneGap y porque preferimos desarrollar con Titanium.

PhoneGap (Apache Cordova)

PhoneGap(http://phonegap.com/) es un entorno para la creación de aplicaciones móviles haciendo uso de HTML, CSS y Javascript, se ejecutan dentro en un componente WebKit del móvil. Este sitema creado por Adobe Systems soporta las plataformas Iphone/Ipad y Android, Palm, Symbian, WebOS, W7 y BlackBerry.

Provee una serie de librerías Javascript desarrolladas en el lenguaje específico de cada plataforma (Java, Objective C, etc.) que permiten acceder a las funcionalidades propias del teléfono. Dependiendo del SO para el que desarrollemos dan acceso al acelerómetro, cámara, agenda de contactos, brújula, archivos, geolocalización, media, red, notificaciones y almacenamiento.

Tiene acceso al DOM, requiere diseñar la aplicación web con componentes o bien hacer uso de un framework web mobile como jQuery Mobile. Realmente son una serie de páginas web con acceso a la mayoría de Apis del móvil que están empaquetadas dentro de una aplicación móvil donde se visualizan con un navegador web.

Para trabajar con cada plataforma hay que usar un sistema distinto: para Iphone/Ipad es necesario usar Xcode y para Android se debe usar Eclipse en ambos casos con una plantilla de proyecto específica.

Por tanto vemos que ambas tecnologías son gratuitas y ambas utilizan HTML, CSS y JavaScript para el desarrollo consiguiendo soportar la mayoría de funcionalidades propias del teléfono y siendo sencillas de programar.
Es una importante ventaja de PhoneGap el hecho de que soporta más tecnologías móviles que Appceletaror al ejecutarse dentro del navegador, pero un gran inconvenientes es que el aspecto por tanto irá relaccionado con el framework web utilizado. Existe la posibilidad de usar frameworks HTML móviles para dar apariencia de aplicación nativa a sus componentes pero el rendimiento no es del todo óptimo ya que deben ser interpretados por el motor del navegador al arrancar.

Por tanto elegimos Appcelerator por la posibilidad de crear aplicaciones nativas y no que corran en un navegador como en el resto de casos. Por esto tanto su apariencia como su rendimiento serán los más óptimos.

2.6 Aplicaciones nativas vs Aplicaciones web

En la actualidad se está dando un gran enfrentamiento para ver cual es la forma más sencilla y económica de lograr desarrollar aplicaciones para los diferentes dispositivos móviles. Con la aparición de HTML5 y todas las posibilidades que incorpora el mercado de las aplicaciones móviles sufrió algunos cambios. Muchos son los que se debaten entre un desarrollo más complejo y en varios lenguajes de programación dependiendo del sistema operativo móvil pero logrando una aplicación totalmente en armonía con el dispositivo, o por el contrario reducir costes y buscar un mecanismo multiplataforma para llegar a todos ellos pero perdiendo muchas de sus ventajas. También han ido apareciendo algunas alternativas híbridas entre ambas.

2.6.1 Aplicaciones nativas

Lo más ideal dado un dispositivo parece desarrollar una aplicación nativa pensada totalmente para el mismo sabiendo sus posibilidades y explotándolas de la mejor manera para conseguir el objetivo que se desee.

Lo primero a destacar de las aplicaciones nativas es que permiten explotar al máximo las prestaciones del dispositivo teniendo un control absoluto sobre el mismo. Esto dará una mejor experiencia de usuario más ligada al dispositivo, además la interfaz no tiene que cargarse junto con el resto de datos.

Su canal de distribución facilita mucho su expansión y las posibilidades de marketing y branding. Las compras se harán de una forma cómoda y centralizada mediante la cuenta de usuario del dispositivo sin tener que facilitar los datos de la tarjeta de crédito.

Las notificaciones han cobrado gran relevancia en todos los SO móviles ya que permiten una interacción más directa con el usuario pudiendo informarle de novedades, cualquier información relevante, etc. Solo podremos tener control sobre las Push Notifications si desarrollamos la aplicación de forma nativa.

También es de gran importancia el comportamiento offline, ya sea haciendo uso de las bases de datos que nos proporcionan, almacenamiento en la red con servidor propio, guardar datos primitivos en pares clave-valor, etc.

2.6.2 Aplicaciones web/HTML5

El lenguaje básico de la web ha ido evolucionando a lo largo de más de 20 años ofreciendo en la actualidad (en su unión con CSS3, DOM y JavaScript) una infinidad de posibilidades. Se pueden crear páginas web así como aplicaciones dotadas de una gran velocidad, rendimiento y experiencia de usuario pudiendo expandirse a través de la web

de una forma mucho más sencilla que las aplicaciones de escritorio y no siendo necesaria su instalación en los dispositivos.

Los principales estándares que definen la web (W3C y WHATWG) trabajan a la par en esta especificación, aunque con métodos algo diferenciados. Podemos ver de la mano del primero de ellos toda la información sobre HTML5: http://www.w3.org/TR/html5/.

Se pueden ver algunas de sus diferencias con la versión anterior en: http://webdesign.about.com/od/html5/a/html_5_whats_new.htm

Algunas de sus aportaciones más destacables son:

- Comunicación bidireccional en tiempo real, la cuál da una infinidad de posibilidades en la iteración con el usuario y en la agilidad en general de las aplicaciones o portales web.
- Soporte en gráficos y multimedia, lo cual da una gran libertad para juegos, animaciones y cualquier efecto gráfico o sonoro que se quiera utilizar.
- Drag & Drop nativo. El efecto de arrastrar y soltar tan común en todos los dispositivos es de gran importancia, sobretodo cuando se pretende que las aplicaciones web den mayor sensación de fusión con el dispositivo.
- Geolocalización. También de gran importancia en el desarrollo para smartphones y tablets.
- Acceso a ficheros del sistema.
- Persistencia. Este punto es fundamental a la hora de desarrollar aplicaciones web ya que es fundamental dotarlas de independencia de la red para no dejarlas inservibles cuando nos quedamos sin conexión.

En el siguiente enlace http://slides.html5rocks.com/#landing-slide se puede ver una presentación bastante completa de como implementar de forma técnica estas funcionalidades.

Esta nueva versión de HTML busca entre otras cosas aportar una mayor funcionalidad reduciendo por tanto la necesidad de plugins externos y sobretodo HTML5 debe ser multiplataforma siendo destacable también su visión hacia dispositivos de bajo consumo como son smartphones y tablets.

Un ejemplo de aplicación HTML5 que se ha adaptado bastante bien a las diferentes plataformas móviles es la de Financial Times: http://apps.ft.com/. En la siguiente imágen podemos observar su apariencia desde un iphone:

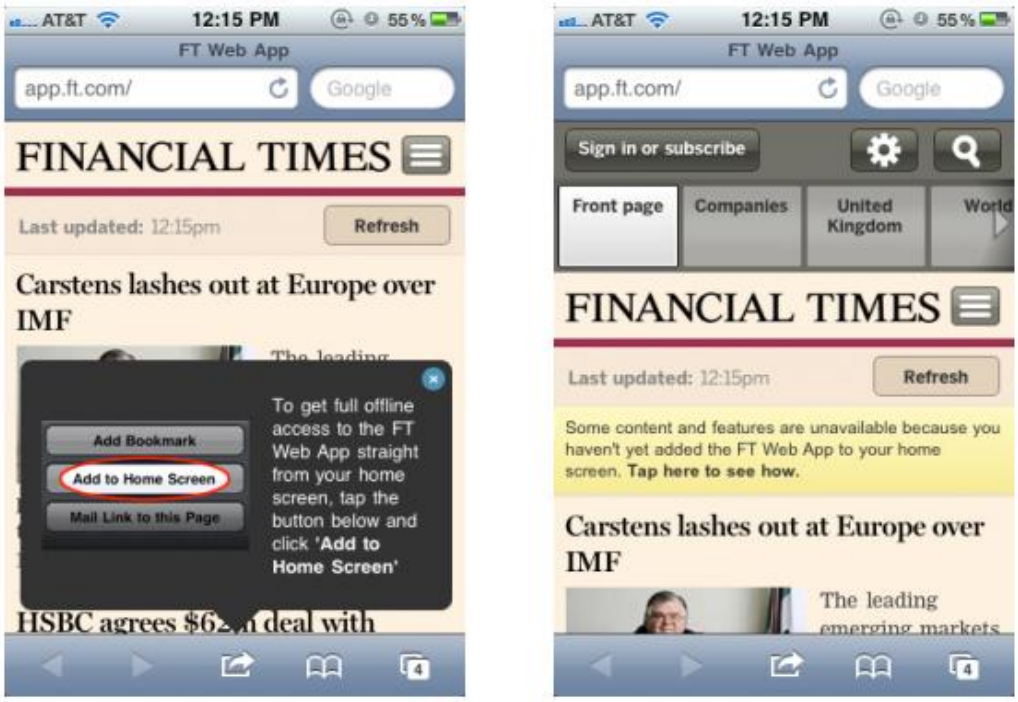


Figura 21. Vista en Iphone de la aplicación web del Financial Times

Por tanto el uso de HTML5 supondrá una serie de ventajas importantes:

La principal relevancia de utilizar aplicaciones web HTML5 es que son multiplataforma, se podrá crear una aplicación una vez y distribuirlo hacia todas las plataformas. Probablemente sean necesarios algunos ajustes para el correcto funcionamiento en todos los dispositivos pero no implicará un desarrollo por cada plataforma.

A parte de programar la aplicación una sola vez, la tecnología es más sencilla que la necesaria para desarrollar aplicaciones nativas, pudiendo utilizar lenguajes como HTML5, JavaScript o CSS. Por tanto está mucho mas extendida y es más sencillo encontrar programadores para estas tecnologías que para las concretas de cada SO móvil.

No hay control de las versiones que se publiquen ni de las posteriores actualizaciones al funcionar con independencia de los canales de distribución de operadores y fabricantes. Las actualizaciones llegan de manera automática a todos los usuarios y plataformas a la vez. Al no estar bajo ningún control hay más libertad a la hora de establecer una política de precios.

Por otro lado tenemos varios inconvenientes:

A pesar de los grandes avances de HTML5 al no estar haciendo un desarrollo nativo con las SDKs y APIs disponibles para cada plataforma estamos inevitablemente perdiendo una parte importante de la funcionalidad del dispositivo así como en apariencia, siendo complicado que la interfaz se sienta tan ligada al teléfono como en casos nativos.

También puede ser complicado lograr un solo diseño que se adapte perfectamente a todos los dispositivos a la vez.

A pesar de la libertad que da no estar controlado por los canales de distribución de cada SO móvil se esta perdiendo un mecanismo de expansión muy fuerte. Si que existen algunas tiendas de distribución HTML5 pero dispersas y poco conocidas.

En un principio requerimos de conectividad en el dispositivo para acceder a este tipo de aplicaciones. Se puede utilizar el manifest.cache para almacenar datos localmente pero probablemente no funcionará adecuadamente en todos los dispositivos y navegadores y además existe una limitación de memoria de 5 MB.

El método de cobro supone un problema para estas aplicaciones, se pierden las ventajas de un sistema de centralización de los mismos. Puede ser muy negativo tanto por la integración de pagos que puede dar muchos problemas en función de los diferentes países, etc. como la barrera que supone conseguir la confianza directa del usuario a la hora de introducir los datos de su tarjeta de crédito.

2.6.3 Hibridas

En muchas ocasiones es útil implementar soluciones híbridas que consisten en aplicaciones nativas en las que determinadas secciones muestran contenido web cargado desde un servidor o desde dentro de la propia aplicación.

Esto permite en determinados casos beneficiarse de las ventajas de facilidad de implementación y actualización de las aplicaciones web, manteniendo las ventajas de potencia y canal de distribución de las aplicaciones nativas.

El problema también de estas aplicaciones es el estricto control que tiene Apple sobre toda aplicación que se intenta subir al App Store. Toda aplicación que contenga un exceso de contenido web no pasará este filtro.

Dado el interés en poder desarrollar aplicaciones multiplataforma, con lenguajes más comunes y aprovechando las ventajas que supone la aplicación nativa y sus tiendas de aplicaciones se han ido desarrollando varias herramientas para facilitar este tipo de aplicaciones. Algunas son Titanium Appcelerator y PhoneGap de las cuales tenemos más información en el apartado 2.5.2.

2.6.4 Comparativa

Es evidente que el coste que supone el desarrollo nativo será considerablemente más alto, por lo que es bueno analizar el tipo de aplicación antes de decidir por qué tipo de aplicación decantarse. Si la aplicación tiene mucho contenido web y no necesita un control minucioso de las funciones del dispositivo se puede valorar el crear una aplicación web HTML5 o bien híbrida. Con esto en vez de tener que desarrollar una app distinta para cada plataforma móvil, utilizando lenguajes distintos y necesitando a personal distinto especializado en cada sistema operativo, podrían desarrollar una sola app HTML5 que valiese para todas las plataformas. Además, el proceso de actualización de esa app es también más sencillo y rápido, pudiendo hacerse directamente y sin tener que esperar la aprobación de la tienda en cuestión.

A pesar de esto será muy difícil que las aplicaciones web alcancen un potencial tan grande que realmente pueda desbancar a las nativas, vemos una serie de puntos a destacar en las aplicaciones nativas frente a HTML5:

- Si lo que deseamos es una aplicación potente con acceso a muchas funcionalidades del teléfono y de una forma lo más eficiente posible habrá que desarrollar de forma nativa, haciendo una aplicación por cada sistema operativo.

- Tampoco se puede olvidar que la persistencia también estará más limitada en HTML5.

- Con HTML5 obtendremos interfaces más pobres y menos ligadas al teléfono. Uno de los puntos fuertes de las aplicaciones nativas es la incorporación de gestos multitouch que dan mayor usabilidad.

- Estar en una tienda como la App Store de Apple o la Google Play Store da a la app muchas más posibilidades de llegar al usuario. Este punto podría superarse en aplicaciones híbridas, aunque en el caso de iOS y el App Store deberá cumplir con todos sus requisitos, no siendo esto tarea fácil sobretodo si se esta integrando una cantidad considerable de contenido web no revisable.

- A pesar de que en ciertos casos se pueda valorar HTML5 como una buena opción dadas nuestras circunstancias no se debe olvidar que es muy difícil que algún día lleguen a estar a la par dada la constante evolución de estos sistemas operativos móviles. No todos los dispositivos soportan de la misma forma las mismas cosas por lo que la idea de hacer la misma cosa para todas ellas siempre se quedará detrás de un desarrollo totalmente pensado para ese dispositivo.

En el presente proyecto se ha decidido hacer una versión nativa para Android así como una híbrida para iOS, pudiendo comparar de una forma directa los resultados en ambos casos. La aplicación híbrida se decide hacer en Titanium ya que, estudiadas Titanium y Phonegap, como las alternativas más adecuadas, vemos que Titanium genera aplicaciones

nativas, utilizando directamente la interfaz propia de iOS y no ejecutándose dentro de un navegador.

Podemos ver algunas comparativas entre aplicaciones nativas, híbridas y HTML5 en:

- http://www.idt.mdh.se/kurser/ct3340/ht12/MINICONFERENCE/FinalPapers/ircse11_submission_16.pdf
- http://cacm.acm.org/magazines/2011/5/107700-mobile-application-development/fulltext
- http://publications.theseus.fi/bitstream/handle/10024/43719/120516_OIo05_Thesis.pdf?sequence=1

3. Análisis del sistema

En este apartado se llevará a cabo un análisis minucioso de la aplicación a desarrollar, centrándose en los objetivos que debe cumplir. Estos objetivos serán la base de todo el desarrollo, a través de ellos se estudiarán las mejores vías para alcanzarlos y servirán de guía principal durante todo el proceso. Por tanto identificaremos el entorno tecnológico para identificar los requisitos de software así como los de usuario.

3.1 Descripción general

Esta aplicación ofrecerá a los usuarios un entorno móvil amigable y reducido para acceder a gran parte de la información y los recursos de la web de Movilforum (programa de partners de Telefónica): http://espana.movilforum.com/. Dado que en la actualidad la mayor parte de los accesos a internet se hacen desde dispositivos móviles es de gran interés tener nuestra representación en los mismos, brindando la posibilidad de acceder a lo más relevante del portal con una interfaz adaptada a este tipo de dispositivos evitando el incómodo acceso desde el navegador.

Teniendo en cuenta las cifras en cuota de mercado de los diferentes sistemas operativos móviles se decide centrarse para el desarrollo del presente proyecto en Android e iOS para cubrir al mayor rango posible de dispositivos.

Esta aplicación debe por tanto desarrollarse tanto para Android como para iOS teniendo en cuenta sus particularidades. En ambos casos debe tener un correcto funcionamiento tanto para su versión móvil como para Tablet.
Se debe tener en cuenta que esta aplicación ocupará un determinado espacio en memoria al instalarlo y posteriormente al ir guardando los datos en memoria para dotar de persistencia a la aplicación.

- En primer lugar se quiere tener acceso a la información que presenta a Movilforum, con un apartado de ¿Qué es?, otro de Beneficios y finalmente otro que de acceso al perfil de Movilforum en las diferentes redes sociales como son Linkedin, Facebook y Twitter.
- En segundo lugar se pretende mostrar el catálogo de Movilforum, para que se puedan consultar las diversas Soluciones y Casos de éxito.
- En tercer lugar un acceso a las Noticias y futuros eventos.
- Acceso al Blog de Movilforum. Acceso externo ya que ya existe una versión del Blog apta para dispositivos móviles: http://blog.movilforum.com/.
- Siempre visible un formulario de contacto.

- Siempre visible una opción para recomendar el contenido que se desee a través del correo electrónico, así como en Facebook y Twitter.
- Siempre visible un formulario para registrarse como miembro de Movilforum.
- Para el correcto funcionamiento sería deseable una conexión internet bien sea 3G o Wifi no siendo esto del todo restrictivo.
- Aparte es importante dotar a la aplicación de un comportamiento offline de tal forma que la parte visualizada de la información quede guardada en estos casos. En caso de tener conexión la información se irá actualizando periódicamente.

La web de la que obtenemos la información ha sido creada con Wordpress por lo que buscaremos un modo de acceder a este Wordpress para recuperar la información que mostraremos en la aplicación.
Por otra parte los formularios de contacto y registro son widgets de Codeeta por lo que veremos también como recuperarlos para mostrarlos en la aplicación.
Estudiamos diversos formatos de comunicación para ver cual será el más adecuado a nuestras necesidades, para recuperar la información de la forma más óptima posible

3.2 Identificación del entorno tecnológico

Tras analizar las necesidades a cubrir con este proyecto se ha optado por definir que entorno necesitará tanto desarrollador como usuario

a) Android

Entorno tecnológico de usuario	
Versión	Android 2.2 o superior

Tabla 3. Entorno tecnológico usuario Android

Accesible tanto desde teléfono móvil como Tablet. No definimos requisitos de memoria ya que ni la aplicación al ser instalada (613 KB) ni la información que puede llegar a guardar son cifras demasiado elevadas.

Entorno tecnológico del desarrollador	
Equipo	Windows 7 Home Premium Intel Core i5 480M 2.67 GHz 4GB RAM
IDE	Eclipse Helios + Plugin Android
SDK Android	2.2 o superior

Dispositivo móvil	- Sony Ericcson Xperia Neo V(Android 2.3.4) - Nexus One(Android 2.2)→Emulador Eclipse - Samsung Galaxy Note(Android 4.0)

Tabla 4. Entorno tecnológico desarrollador Android

Siendo conscientes de la fragmentación existente en Android se ha decidido programar para la versión 2.2 y superiores debido a que las versiones anteriores están en periodo de desaparición, suponiendo entre las versiones 1.5 (Cupcake), 1.6 (Donut) y 2.1 (Eclair) un 6.1% de la cuota de mercado total. Principalmente se desarrolla sobre la versión 2.3.4 siendo Gingebread la versión predominante con un 64% de la cuota. A pesar de lanzarse la versión 4.0 (Ice Cream Sandwich) a finales de 2011 no ha conseguido desplazar a la anterior.

b) iOS

Entorno tecnológico de usuario	
Versión	iOS 4 o superior

Tabla 5. Entorno tecnológico de usuario iOS

La versión iOS 4 es compatible con el iPhone 4, el iPhone 3GS, y el iPhone 3G, y con los iPod touch de segunda y tercera generación

Espacio memoria: La aplicación al ser descargada ocupa 3.33 MB más en ejecución lo necesario para dotar a la aplicación de persistencia. No serán cifras demasiado considerables.

Entorno tecnológico del desarrollador	
Equipo	Mac OS X Versión 10.7.4 Intel Core 2 Duo 4GB RAM 1067 MHz DDR3
IDE	Titanium Studio 2.0.1.201204132053
SDK iOS	4 o superior
Dispositivo móvil	- Iphone 4S(iOS5)→Emulador Titanium - Iphone 3GS(iOS5) - Ipad 2

Tabla 6. Entorno tecnológico desarrollador iOS

La última versión estable de iOS es la 5.1.1. A principios de Junio iOS presentaba su nueva versión 6.0 aunque le quedan unos meses para salir al mercado, probablemente

acompañado de un nuevo modelo de iPhone. Adaptamos la aplicación para que funcione para iOS5 ya que en iOS normalmente el usuario tendrá el dispositivo actualizado a la última versión aunque no goce de todas las novedades de la misma. Para algunos dispositivos las nuevas versiones directamente dejan de ser compatibles.

3.3 Casos de uso

El objetivo de este punto es identificar los requisitos funcionales del sistema en función de la interacción del usuario. Se describen por tanto las interacciones típicas usuario-sistema viendo las posibilidades que deberá tener el mismo. Se utiliza el modelo de casos de uso de Ivar Jacobson:

http://www-2.dc.uba.ar/materias/isoft1/2001_2/apuntes/CasosDeUso.pdf

a) Descripción textual

Cada caso de uso cuenta con los siguientes atributos:

- **Título:** descripción mínima del caso de uso.
- **Objetivo:** funcionalidad requerida.
- **Actores:** agente externo que interactúa con el sistema.
- **Precondiciones:** estado del sistema antes de la ejecución del caso de uso.
- **Postcondiciones:** estado del sistema después de la ejecución del caso de uso.
- **Escensario principal:** interacción con el escenario viendo los cambios que se producen en el mismo.
- **Escenarios alternativos:** acciones alternativas al escenario principal.

Título	Acceso formulario contacto
Objetivo	Permitir al usuario contactar con Movilforum
Actores	Usuario de la aplicación
Precondiciones	Aplicación instalada en el terminal
Postcondiciones	El usuario accede a la opción “Contacto” del menú inferior
Escenario principal	El usuario accede a la aplicación, se muestra la pantalla de bienvenida y posteriormente la Home, desde la mayoría de pantallas a las que accederá desde este punto tendrá la opción “Contacto” en un menú en la parte inferior. El usuario toca el icono de “Contacto” y accede a un formulario con los siguientes campos: - Nombre - Apellidos - Correo electrónico - Teléfono

	- Empresa - Tipo de consulta(Soporte técnico, Sobre el programa de partners, Genérica, Evento Nokia Windows Phone) - Consulta
Escenarios alternativos	Si no se rellenan todos los campos el sistema informa al usuario para que los rellene Una vez los datos están rellenados correctamente y damos al botón Enviar: - Si hay conexión se envía y nos informa del envió satisfactorio. - Si no hay conexión no se envía informándonos de la falta de conectividad.

Tabla 7. Acceso formulario contacto

Título	**Acceso formulario registro**
Objetivo	Permitir al usuario unirse a Movilforum
Actores	Usuario de la aplicación
Precondiciones	Aplicación instalada en el terminal
Postcondiciones	El usuario accede a la opción “Únete” del menú inferior
Escenario principal	El usuario accede a la aplicación, se muestra la pantalla de bienvenida y posteriormente la Home, desde la mayoría de pantallas a las que accederá desde este punto tendrán la opción “Únete” en un menú en la parte inferior. El usuario toca el icono de “Únete” y accede a un formulario con los siguientes campos: - Nombre(Nombre de usuario, Nombre, Apellidos, Alias) - Información de contacto(Email, Sitio Web, Empresa) - Acerca de ti(Contraseña, Repetir contraseña)
Escenarios alternativos	Si no se rellenan todos los campos el sistema informa al usuario para que los rellene Una vez los datos están rellenados correctamente y damos al botón Enviar: - Si hay conexión se envia y nos informa del envió satistactorio. - Si no hay conexión no se envía informandonos de la falta de conectividad.

Tabla 8. Acceso formulario registro

Título	Acceso Catálogo
Objetivo	Mostrar Catálogo
Actores	Usuario de la aplicación
Precondiciones	Aplicación instalada en el terminal
Postcondiciones	El usuario accede a la opción "Catálogo" de la Home
Escenario principal	El usuario accede a la aplicación, se muestra la pantalla de bienvenida y posteriormente la Home, en la cuál una de las opciones es "Catálogo". El usuario toca el icono de "Catálogo" y accede a una nueva pantalla donde tenemos dos nuevas opciones: - Soluciones. Si el usuario pulsa esta opción de abre un diálogo con las siguientes opciones: Fuerzas de Venta, Fuerzas de Campo, Loc/Gest. Flotas, M2M, Domótica y Telecontrol y Marketing Móvil. Al pulsar sobre cada una de ellas se abre una nueva pantalla con una lista del contenido de la opción seleccionada. Todas los posts que se muestran en esta lista tendrán una opción "Leer más…" que nos llevará a una nueva pantalla donde podremos ver el texto completo. - Casos de éxito. Si el usuario pulsa esta opción de abre un diálogo con las siguientes opciones: Educación, Sanidad y Farma, Seguridad, Cultura, Ocio y Turismo, Trámites, Media y Publicidad, Energía, Construcción, Automoción, Distr., Trans.. y Logística, Administración Pública, Bancos y Cajas y Aseguradoras. Al pulsar sobre cada una de ellas se abre una nueva pantalla con una lista del contenido de la opción seleccionada. Todas los posts que se muestran en esta lista tendrán una opción "Leer más…" que nos llevará a una nueva pantalla donde podremos ver el texto completo.
Escenarios alternativos	El sistema nos informará si nos encontramos sin conexión de red y no se puede recuperar el contenido. Cuando la aplicación esta actualizando o descargando por primera vez datos se nos mostrará un mensaje que dice:"Cargando…".

Tabla 9. Acceso catálogo

Título	Acceso Noticias y Eventos
Objetivo	Mostrar Noticias y Eventos
Actores	Usuario de la aplicación
Precondiciones	Aplicación instalada en el terminal
Postcondiciones	El usuario accede a la opción "Noticias y Eventos" de la Home
Escenario principal	El usuario accede a la aplicación, se muestra la pantalla de bienvenida y posteriormente la Home, en la cuál una de las opciones es "Noticias y Eventos". El usuario toca el icono de "Noticias y Eventos" y accede a una nueva pantalla donde tenemos dos opciones: - Noticias. Al pulsar sobre esta se abre una nueva pantalla con un listado de fragmentos de noticias. Todas las noticias que se muestran en esta lista tendrán una opción "Leer más…" que nos llevará a una nueva pantalla donde podremos ver el texto completo de la noticia. - Eventos. Al pulsar sobre esta se abre una nueva pantalla con un listado de eventos de los que se indica el Nombre, la Fecha y el Lugar de celebración. Al pinchar sobre cualquiera de los eventos de la lista accederemos a una nueva pantalla donde podremos ver la descripción del evento.
Escenarios alternativos	El sistema nos informará si nos encontramos sin conexión de red y no se puede recuperar el contenido. Cuando la aplicación esta actualizando o descargando por primera vez datos se nos mostrará un mensaje que dice:"Cargando…".

Tabla 10. Acceso noticias y eventos

Título	Acceso Conoce MF
Objetivo	Mostrar Conoce MF
Actores	Usuario de la aplicación
Precondiciones	Aplicación instalada en el terminal
Postcondiciones	El usuario accede a la opción "Conoce MF" de la Home
Escenario principal	El usuario accede a la aplicación, se muestra la pantalla de bienvenida y posteriormente la Home, en la cuál una de las opciones es "Conoce MF". El usuario toca el icono de "Conoce MF" y accede a una nueva pantalla donde tenemos tres nuevas

	opciones: - Qué es. Si el usuario pulsa esta opción se abre una nueva pantalla que muestra una descripción de lo que es Movilforum. - Beneficios. Si el usuario pulsa esta opción se abre una nueva pantalla que muestra los beneficios de Movilforum. - RRSS. Si el usuario pulsa esta opción se abre un diálogo con las opciones: Linkedin, Facebook, Twitter que darán acceso al perfil de Movilforum en cada una de ellas.
Escenarios alternativos	El sistema nos informará si nos encontramos sin conexión de red y no se puede recuperar el contenido. Cuando la aplicación esta actualizando o descargando por primera vez datos se nos mostrará un mensaje que dice:"Cargando…".

Tabla 11. Acceso Conoce MF

Título	**Acceso Blog Movilforum**
Objetivo	Mostrar Blog
Actores	Usuario de la aplicación
Precondiciones	Aplicación instalada en el terminal
Postcondiciones	El usuario accede a la opción "Blog"
Escenario principal	El usuario accede a la aplicación, se muestra la pantalla de bienvenida y posteriormente la Home, en la cuál una de las opciones es "Blog". El usuario toca el icono de "Blog" y se abre el navegador mostrando la versión reducida para el Blog de Movilforum.
Escenarios alternativos	El sistema nos informará si nos encontramos sin conexión de red y no se puede abrir la url indicada.

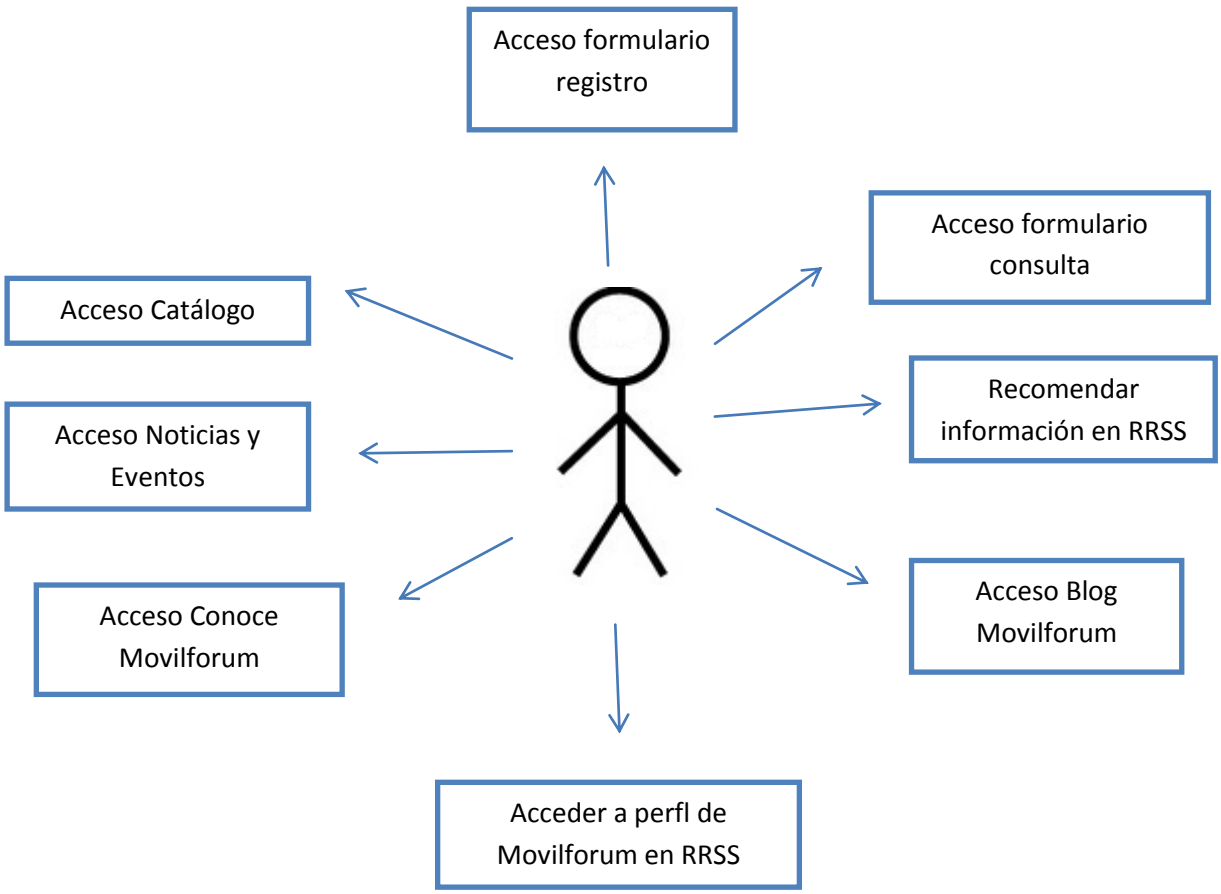
Tabla 12. Acceso Blog MF

Título	**Recomendar contenido Movilforum en RRSS**
Objetivo	Recomendación de contenidos en RRSS
Actores	Usuario de la aplicación
Precondiciones	Aplicación instalada en el terminal
Postcondiciones	El usuario accede a la opción "Recomendar"
Escenario principal	El usuario accede a la aplicación, se muestra la

	pantalla de bienvenida y posteriormente la Home, desde la mayoría de pantallas a las que accederá desde este punto tendrá la opción "Recomendar" en un menú en la parte inferior. El usuario toca el icono de "Recomendar" y se abre un diálogo con las opciones: Email, Publicar en twitter y Movilforum en facebook. Dependiendo de la pantalla donde el usuario se encuentre recomendará el contenido que el usuario este viendo en ese momento. En el caso del email se abrirá un mensaje con un asunto y un contenido donde podremos editar la información y poner el destinatario del mensaje. En facebook y twitter aparecerá el mensaje en el tabón para su publicación pudiendo también editarlo.
Escenarios alternativos	El sistema nos informará si nos encontramos sin conexión de red y no se puede abrir la url indicada.

Tabla 13. Recomendar contenido MF en RRSS

b) Descripción gráfica



3.4 Requisitos Software

Tras analizar el proyecto propuesto se han definido una serie de requisitos que dan una visión completa de su funcionalidad. Los veremos en forma de tablas con las siguientes propiedades:

- Descripción: Especifica el significado del requisito.

- Tipo: funcionales, de rendimiento, de usabilidad, de restricción
- Estabilidad: Hace referencia a la sensibilidad pudiendo tener los valores 'estable' o 'no estable'.
- Prioridad: Define la prioridad con la que debe ser resuelto dicho requisito. Puede tomar los valores de "Alta", "Media" y "Baja".

Descripción	Acceso adaptado al contenido más relevante de la página web de Movilforum: - Conoce MF(Qué es, Beneficios) - Catálogo(Soluciones y Casos de Éxito) - Noticias y Eventos
Tipo	Funcional
Estabilidad	Estable
Prioridad	Alta

Tabla 14. Acceso contenido relevante

Descripción	Acceso al perfil de Movilforum en redes sociales (Linkedin, Facebook, Twitter)
Tipo	Funcional
Estabilidad	Estable
Prioridad	Alta

Tabla 15. Acceso perfil MF en RRSS

Descripción	Enlace en el menú principal a la versión móvil del Blog de Movilfo-rum
Tipo	Funcional
Estabilidad	Estable
Prioridad	Alta

Tabla 16. Enlace en el menú principal a la versión móvil del Blog de Movilforum

Descripción	Posibilidad de contactar con Movilforum mediante un formulario con los siguientes campos: - Nombre - Apellidos - Correo electrónico - Teléfono - Empresa - Tipo de consulta(Soporte técnico, Sobre el programa de partners, Genérica, Evento Nokia Windows Phone) - Consulta
Tipo	Funcional
Estabilidad	Estable
Prioridad	Alta

Tabla 17. Formulario contacto MF

Descripción	Posibilidad de unirse al programa de Movilforum mediante un formulario con los siguientes campos: - Nombre(Nombre de usuario, Nombre, Apellidos, Alias) - Información de contacto(Email, Sitio Web, Empresa) - Acerca de ti(Contraseña, Repetir contraseña)
Tipo	Funcional
Estabilidad	Estable
Prioridad	Alta

Tabla 18. Formulario unión programa de partners

Descripción	Permitir recomendar la información de las diversas categorías(Email, Twitter, Facebook)
Tipo	Funcional
Estabilidad	Estable
Prioridad	Alta

Tabla 19. Recomendar en RRSS

Descripción	Tiempo de respuesta máximo de 5 segundos al actualizar o acceder por primera vez a los datos mostrando un mensaje de progreso de la carga
Tipo	De rendimiento
Estabilidad	Estable
Prioridad	Alta

Tabla 20. Tiempo de respuesta y mensaje de progreso

Descripción	Responder adecuadamente a problemas de conexión y errores que impidan el correcto funcionamiento de la aplicación
Tipo	De rendimiento
Estabilidad	Estable
Prioridad	Alta

Tabla 21. Respuesta a errores

Descripción	Ofrecer un comportamiento offline al usuario pudiendo consultar sin conexión toda la información ya consultada con anterioridad
Tipo	De rendimiento
Estabilidad	Estable
Prioridad	Media

Tabla 22. Comportamiento offline

Descripción	Actualización periódica del contenido cada 6 horas o cada 24 horas dependiendo del tipo de contenido. - Conoce MF(Qué es, Beneficios)→24h - Catálogo(Soluciones y Casos de Éxito)→6h - Noticias y Eventos→6h - Formularios(Contacto, Únete) →24h
Tipo	De rendimiento
Estabilidad	Estable
Prioridad	Media

Tabla 23. Actualización periódica de contenido

Descripción	Interfaz intuitiva y atractiva para el usuario, reducir al máximo el número de interacciones del usuario para llegar a toda la información disponible en la aplicación
Tipo	De usabilidad
Estabilidad	Estable
Prioridad	Alta

Tabla 24. Interfaz intuitiva y atractiva

Descripción	Versión mínima para ejecutar la aplicación - Android: 2.2 - iOS: 4
Tipo	De restricción
Estabilidad	Estable
Prioridad	Alta

Tabla 25. Versión mínima

Descripción	Deberá haber conexión bien sea Wifi o 3G para un funcionamiento óptimo y completo de la aplicación
Tipo	De restricción
Estabilidad	Estable
Prioridad	Alta

Tabla 26. Formulario unión programa de partners.

En este punto se define el diseño de la aplicación, una vez hecho un análisis previo del sistema y antes de comenzar su implementación es necesario pasar por esta fase. Se trata desde un diseño a nivel arquitectónico hasta su apariencia final.

4.1 Arquitectura de la aplicación

Esta aplicación se basa en un modelo cliente-servidor, donde el cliente hace peticiones al servidor y este devuelve una respuesta.

El **modelo cliente-servidor** es un modelo de aplicación distribuida en la que un prestador de servicio, el servidor, responde a las peticiones de los clientes. Una máquina servidora es una máquina que está ejecutando uno o más programas servidores que comparten sus recursos con los clientes. Un cliente no comparte sus recursos sino que solicita al servidor su contenido o servicio. Por tanto son los clientes quienes inician el proceso de comunicación.

El sistema se comporta como un cliente-servidor ya que es un sistema distribuido en el que son los clientes los que realizan las peticiones a la API. El servidor se limita a responder estas peticiones aportando los datos solicitados.

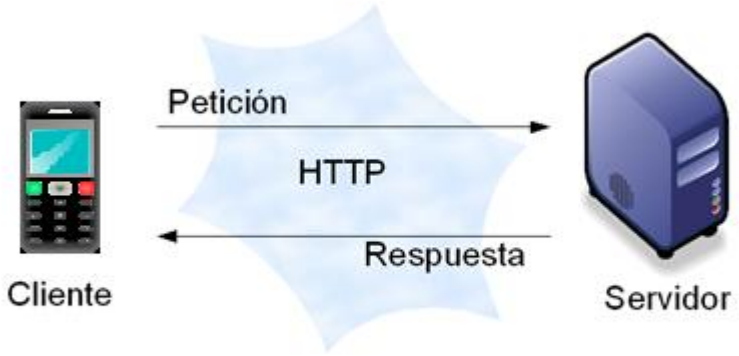


Figura 22. Arquitectura cliente-servidor

En este caso el cliente será el terminal móvil sobre el que corre la aplicación y que realizará las peticiones a través de internet por el protocolo HTTP.

El **protocolo http** (Hyper Text Transfer Protocol) es un protocolo del nivel de aplicación para sistemas de información multimedia distribuidos. Entre las propiedades de este protocolo se pueden destacar:

- Se utiliza URI (Universal Resource Identifier) para localizar sitios (URL) o nombres (URN) sobre los que se aplicará una petición.

- Arquitectura cliente-servidor: Este protocolo se ajusta al paradigma petición/respuesta. La comunicación se establece sobre el protocolo TCP/IP.
- Protocolo no orientado a conexión. Una vez que el servidor responde a una petición realizada desde el cliente se interrumpe la conexión entre ellos.
- HTTP utiliza tipos MIME (Multipart Internet Mail Extension) para determinar el tipo de datos que transporta. Una vez que el servidor http envía la respuesta a una petición desde el cliente, incluye una cabecera que le indica el tipo de datos que la componen, siendo el cliente el encargado de gestionar esos datos.

HTTPS es la versión segura del HTTP, este sistema usa un cifrado SSL/TLS para crear un canal cifrado para la información. Muchas veces se necesita envíar información de gran confidencialidad y un protocolo HTTP simplemente podría suponer un gran riesgo. En el caso del servidor de Codeeta vemos que las peticiones son de este tipo.

En este caso es necesario hacer peticiones por un lado a Codeeta y por otro al Wordpress de Movilforum.

Codeeta tiene implementada ya una API que devuelve respuestas en formato JSON por lo que utilizaremos este formato de intercambio de datos.

Por otra parte después de analizar diversos formatos de comunicación seleccionamos también JSON para las peticiones a Wordpress. Utilizaremos un plugin ya implementado que implementa una API JSON sobre nuestro Wordpress. Añadiendo métodos a las url's podremos extraer toda la información que deseemos.

JSON es fácil de comprender, crear e interpretar. Los lenguajes han ido incorporando funciones nativas que nos permiten trabajar con JSON de una forma muy sencilla. Además JSON es nativo de JavaScript, son prototipos de datos los cuales tiene propiedades que pueden ser accedidas utilizando el simplemente la estructura "objeto.propiedad", por lo cual, extraer datos de un objeto JSON es muy fácil comparado con otras opciones como XML.

Almacenamiento de datos.

Necesitaremos almacenar de algún modo información ya que deseamos dotar a la aplicación de persistencia para que no tenga una dependencia absoluta de la conexión existente en un momento dado y el usuario pueda consultar posteriormente la información que ya ha consultado con anterioridad.

No usaremos una base de datos directamente dado que no será una cantidad demasiado grande de información y sobretodo la estructura de la información no será compleja ya que lo que realmente queremos guardar son pares clave-valor, siendo el valor una respuesta JSON y la clave una intuitiva para reconocer la información que estamos guardando.

Viendo las opciones de almacenamiento de datos que propone Android tenemos:

SharedPreferences: permite guardar y recuperar pares clave-valor de los tipos de datos primitivos (booleans, floats, ints, longs, y strings). Estos datos se conservan aunque se cierre la aplicación.

En el caso de iOS en Titanium, como alternativa a las bases de datos tenemos algo muy similar al SharedPreferences de Android:

Properties que se utiliza para guardar datos de la aplicación en pares clave-valor y también con persistencia.

Modelo para Aplicación Movilforum

El usuario interactúa con la aplicación móvil mediante su interfaz requiriendo tipos concretos de información.

La aplicación móvil captura esta acción que podrá ser de tres tipos:

- **Llamada a la API de Codeeta**. En este caso convertirá la interacción en una llamada a esta API. Serán de este tipo los casos de "Contacto" y "Únete".

  Por ejemplo si pulsamos la opción Contacto la llamada realizada será:

  "https://api.codeeta.com/"+*api_token*+"/widgets/"+*widgetId_contacto*;

  Siendo api_token y widgetId_contacto dos identificadores que indican el usuario de Codeeta que está accediendo y el widget que esta solicitando respectivamente.

  Esta petición se envía al servidor a través de HTTP y el servidor devuelve la estructura JSON del formulario.

  Esta respuesta es muy fácil de interpretar, bastará con ir obteniendo los campos del formulario y pintándolos en la aplicación móvil.

- **Llamada a la API JSON de Wordpress.** En este caso convertirá la interacción en una llamada a esta API. Serán de este tipo todas las peticiones dentro de "Catálogo", "Noticias/Eventos" y "Conoce MF".

  Por ejemplo si pulsamos la opción Noticias la llamada realizada será:

  http://espana.movilforum.com/api/get_category_posts/?id=7&count=5

Y la respuesta que mandará el servidor a esta petición será una estructura JSON con los últimos 5 posts existentes en la categoría noticias (que tendrá el id=7 en Wordpress).
Extraeremos de este JSON los campos “title”, “content”, “excerpt”, “url” y “date”. El campo “excerpt” es un fragmento del contenido que nos será de gran utilidad para mostrar inicialmente un listado de noticias en este caso de las que solo se verá un pequeño fragmento, accediendo únicamente a la totalidad del contenido cuando pulsemos al enlace “Ver más…”. El campo “url” lo necesitaremos para poder hacer recomendaciones que consistirán en mandar esta url con un mensaje bien sea a través de correo electrónico, de Facebook o de twitter. “date” lo extraeremos concretamente en el caso de noticias para visualizar la fecha en el listado de noticias.

Para el resto de peticiciones a Wordpress según estemos ante Páginas o ante entradas categorizadas como en el caso de noticias, o en el caso de Eventos que es un tipo de contenido personalizado, las peticiones variarán basándose en la API y las respuestas también serán algo diferentes. Esto se verá más detenidamente en el apartado de Implementación.

- **Interacción con navegador web**, ya sea queriendo visualizar el perfil de Movilforum en las RRSS, como recomendando un contenido concreto de la aplicación o accediendo al Blog. La interacción se convertirá en una llamada al navegador con la dirección y datos necesarios.

4.2 Interfaz de usuario

Vemos un diseño de como se distribuirán las pantallas de esta aplicación:

Para un máximo aprovechamiento de la pantalla y dadas las funciones de nuestra aplicación forzaremos la orientación vertical de la misma.

1. IDEAS DE DISEÑO Y CONCEPTO

La idea es tener una aplicación que cumpla los objetivos con la máxima sencillez.

El objetivo es dar la máxima difusión al programa de partners, dándose a conocer de la forma más atractiva posible.

Esta aplicación está orientada a dos tipos de público, prioritariamente a clientes y posibles clientes, y por otro lado a empresas que pertenezcan al programa.

Se desea un diseño lo más limpio posible, siguiendo la línea de la página web:

Figura 23. Web Movilforum

Se pretende conseguir la máxima usabilidad, buscando que el usuario se sienta en todo momento ubicado sin perder el hilo de la navegación.

Es importante tener en cuenta que será una app en constante actualización ya que muchos de los contenidos (Soluciones, Casos de éxito, Noticias, Eventos..) irán variando prácticamente a diario. Por tanto se debe estructurar correctamente la aplicación para una correcta puesta a punto.

A continuación, está el árbol de contenido/navegación que se puede tomar como punto de partida para construir la app. A medida que se vayan necesitando más contenidos, se irán suministrando.

2. ÁRBOL DE NAVEGACIÓN/ CONTENIDOS

BLOQUE 1: Conoce MF

- ¿Qué es?
- Beneficios

- RRSS: enlaces a nuestros perfiles en Twitter, facebook, linkedin)

BLOQUE 2: Catálogo

- Soluciones: Fuerzas de venta, Fuerzas de campo, Localización/ Gestión de Flotas, M2M- Domótica y Telecontrol, Marketing Móvil
- Casos de éxito: educación, sanidad y farma, seguridad, cultura, ocio y turismo, trámites, media y publicidad, energía, construcción, automoción, distribución-transportes y logística, AAPP, Bancos y Cajas, Aseguradoras. (este es el número máximo de ítems que puede haber, pero actualmente hay algunos que no tienen nada en la web- confirmar más adelante).

BLOQUE 3: Noticias (no subiremos todas, sino las que nosotros generemos o nos interesen) y Eventos (calendario)

MENÚ INFERIOR (siempre visible- marcado abajo con un rectángulo rojo)

- Contacto
- Recomendar (email, publicar en twitter, publicar en facebook)
- Únete

En este apartado veremos como se ha ido implementando la aplicación, se centrará en los puntos más importantes ya que no se pretende hacer un tutorial de desarrollo sino ver los puntos más importantes de este desarrollo y las consideraciones tomadas a lo largo de este proceso.

ANDROID

Se han usado varias y diversas tecnologías en el proceso de desarrollo e implementación de esta aplicación. El software que más se ha utilizado durante el desarrollo ha sido el entorno de desarrollo Eclipse Helios 3.6.1, con Java 1.6.0_21 para sistemas de 64 bits. Este entorno es en el que se ha editado todo el código fuente.

A este entorno de desarrollo se le ha instalado el ADT un plug-in para poder desarrollar aplicaciones en Android usando el Eclipse. Este plug-in nos permite crear proyectos Android, crear interfaces de usuario para aplicaciones, añadir componentes basados en el Android Framework API, debuggear aplicaciones Android y exportar APKs, lo que permite distribuir aplicaciones Android. Además nos permite gestionar y crear terminales emulados Android. Para descargarse este plug-in basta con dirigirse a: http://developer.android.com/sdk/eclipse-adt.html.

Para emular el entorno de ejecución de la aplicación (un terminal móvil), se ha utilizado el Android Virtual Device Manager. Este gestor de terminales móviles virtuales permite crear emuladores de terminales con distintas características como la versión del sistema operativo.

iOS

Para la aplicación híbrida en iOS utilizamos el entorno Titanium. Este entorno esta basado en Eclipse por lo que tanto para Android como para iOS el entorno es muy similar. La configuración para empezar a desarrollar con iOS es muy sencilla, bastará con contar con el SDK de iOS. El propio Titanium permite emular terminales Iphone e Ipad.

Comunicación con Wordpress

En primer lugar nos centramos en como hacer las peticiones a Wordpress y una vez recibida la respuesta JSON como tratarla.

Dado el portal Wordpress al que debemos acceder y teniendo acceso al mismo instalamos el plugin JSON API para Wordpress ya que consideramos que el formato óptimo de intercambio de datos será JSON y que tras ver diferentes opciones consideramos este plug-in bastante intuitivo y eficiente.

El plug-in JSON API permite recuperar y manipular el contenido de WordPress mediante solicitudes HTTP. En este caso queremos obtener la información de diversos posts así cómo páginas y tipos de contenido personalizados.

INSTALACION PLUGIN JSON WORDPRESS

Descargamos el plugin de: http://wordpress.org/extend/plugins/json-api/

1. Una vez instalado el plugin de una forma muy sencilla: Sube el json-api carpeta para los wp-content/plugins / o directorios o instalar directamente a través del instalador de plugins.
2. Activar el plugin a través del menú 'Plugins' en WordPress o mediante el enlace proporcionado por el instalador del plugin.

Las solicitudes

Se utiliza un simple estilo REST HTTP GET o POST. Para invocar la API se incluye un valor de consulta en la URL.

JSON API funciona en dos modos:

1. *El modo implícito* devuelve el contenido que se visualiza en la página a la que se esta llamándose

Ejemplos:

- http://www.example.org/?json=1
- http://www.example.org/?p=47&json=1
- http://www.example.org/tag/banana/?json=1

2. *Modo explícito* en este caso se hacen llamadas a métodos de la API. Algunos métodos son: get_recent_posts, get_post, get_page, get_date_posts, get_category_posts, etc.

Ejemplos:

- http://www.example.org/?json=get_recent_posts
- http://www.example.org/?json=get_post&post_id=47
- http://www.example.org/?json=get_tag_posts&tag_slug=banana

En este caso utilizaremos el modo explícito para llevar a cabo las solicitudes ya que aporta una información más específica y responde mejor a las necesidades de esta aplicación.

Las llamadas que realizaremos a la API serán del tipo:

- http://espana.movilforum.com/nuestros-beneficios/?json=get_recent_posts
- http://espana.movilforum.com/api/get_recent_posts/?dev=1&post_type=ai1ec_event&orderby=ai1ec_event_date&order=desc

- http://espana.movilforum.com/api/get_category_posts/?id=331

En el caso de páginas cuyo contenido es estático como el caso de ¿Qué es Movilforum? Simplemente será suficiente con añadir a la url de este apartado: "/?json=get_recent_post".

La segunda url está llamando al tipo de contenido Eventos y ordenándoles por la fecha en orden descendente. En este Wordpress se ha creado un tipo personalizado de post llamado Evento, no tendremos por tanto una página estática a la que acceder sino a un tipo concreto de contenido, por lo que la llamada será diferente. Ponemos la raíz de la url que será: "http://espana.movilforum.com" después accedemos a la api, llamamos a los posts más recientes e indicamos el tipo de post: "post_type=ai1ec_event", y por último ordenamos por la fecha de estos posts: "oderby=ai1ec_event_date" en orden descendente: "order=desc".

Y finalmente en el caso de la tercera url estamos obteniendo los posts por categoría de esta forma diferenciaremos los posts de las diversas categorías dentro de Soluciones, Casos de éxito y Noticias. Esta forma de clasificación es muy sencilla simplemente a cada post que se crea se le asigna una categoría a la wordpress asigna un id. Con esto es muy sencillo obtener los posts de una categoría concreta poniendo la raíz: "http://espana.movilforum.com" llamando después a la api con el método: "/get_category_posts/" y finalmente indicando el id de la categoría que se va a obtener, en este caso: "/?id=331"

La respuesta a las llamadas anteriores será una cadena JSON con todos los posts solicitados. En el caso de la llamada a la página beneficios obtenermos la siguiente respuesta:


```
{"posts":[
{"comment_count":0,
"tags":[],
"status":"publish",
"excerpt":"Nuestros beneficios Ventajas del Programa de Partners de
Telefónica Desarrollo de Negocio: Incorpora tus aplicaciones y
soluciones y aprovecha la fuerza comercial de Telefónica. Da a conocer
tus casos de éxito y experiencias. Contacto con los clientes de
Telefónica y con el mundo empresarial tecnológico, a través de
eventos tecnológicos significativos (Móvil Forum Conference, &hellip; <a
href=\"http:\/\/espana.movilforum.com\/nuestros-beneficios-
movil\/\"><img src=\"\/wp-content\/themes\/IMG_GEN\/leer_mas.png\">
<span class=\"meta-nav\"><\/span><\/a>",
"comment_status":"open",
"date":"2012-05-17 12:04:53",
"type":"page",
"url":"http:\/\/espana.movilforum.com\/nuestros-beneficios-movil\/",
"modified":"2012-05-17 13:44:13",
"content":"<h1>Nuestros beneficios<\/h1>\n<h2>Ventajas del Programa de
Partners de Telefónica Desarrollo de Negocio:<\/h2>\n<ul>\n<li>Incorpora
tus aplicaciones y soluciones y aprovecha la fuerza comercial de
Telefónica.<\/li>\n<li>Da a conocer tus casos de éxito y
experiencias.<\/li>\n<li>Contacto con los clientes de Telefónica y con
el mundo empresarial tecnológico, a través de eventos tecnológicos
significativos (Móvil Forum Conference, SIMO, 3GSM...",
```

```
"id":9767,
"author":{"id":1,"first_name":"","nickname":"admin","description":"","na
me":"admin","last_name":"","slug":"admin","url":""},
"title":"Nuestros beneficios",
"categories":[],"slug":"nuestros-beneficios-movil",
"attachments":[],
"comments":[],
"title_plain":"Nuestros beneficios"}],
"count":1,
"status":"ok",
"count_total":0,
"pages":0
}
```


La siguiente respuesta es la de Fuerzas de Venta, ya no será una página sino una categoría concreta con una serie de posts. La llamada es exactamente de la forma anteriormente vista pero en estos casos tendremos un listado con los diferentes casos de fuerzas de venta pudiendo después abrir el que nos interese dando a "Leer más…", esto nos llevará a una nueva pantalla en la que podremos ver el texto completo del post con sus imágenes y todo su contenido.

Para estos casos nos viene muy bien el campo excerpt que es un extracto del texto del contenido libre de links, imágenes, tablas, etc..por lo que es muy útil para mostrar listados de posts y que luego el usuario seleccione cada post si quiere ver más.

El campo url también en el caso de esta aplicación será de gran interés ya que el usuario en todo momento puede recomendar el contenido que esta visualizando a través de redes sociales y de su correo electrónico por lo que utilizaremos esta url en estos casos.

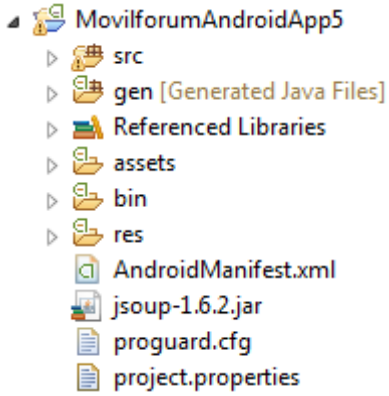

```
{"posts":
[{"
comment_count":0,
"tags":[],
"status":"publish",
"excerpt":"Es una aplicación móvil pensada para aquellas empresas que
quieren  modernizar su proceso de expedición, entregas y recepción de
mercancías, así como mejorar la distribución a sus clientes y obtener un
mayor control de las incidencias ocurridas  durante la misma.",
"comment_status":"open",
"date":"2011-11-23 10:28:54",
"type":"post",
"url":"http:\/\/espana.movilforum.com\/2011\/11\/23\/solucion-movil-de-
entregas-y-recogidas\/",
"modified":"2012-06-11 01:06:24",
"content":"<div>Es una aplicación móvil pensada para aquellas empresas
que quieren  modernizar su proceso de expedición, entregas y recepción
de mercancías, así como mejorar la distribución a sus clientes y obtener
un mayor control de las incidencias ocurridas durante la
misma.<\/div>…",
"id":6570,
"author":
{"id":229,
"first_name":"Vanesa",
"nickname":"vgarcia",
"description":"",
"name":"vgarcia",
```

```
"last_name":"García González",
"slug":"vgarcia",
"url":""},
"title":"Solución móvil de Entregas y Recogidas",
"thumbnail":"http:\/\/espana.movilforum.com\/files\/2011\/11\/entregas.jpg",
…
```

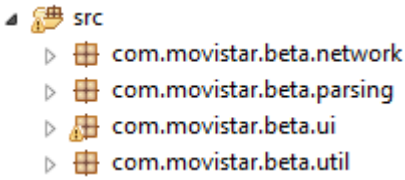

Mas adelantes se ve como acceder a los diferente campos del JSON de forma muy sencilla.

5.1 Implementación aplicación Android:

El árbol que define el proyecto es el siguiente:



- **src**: en este directorio es donde se almacenarán los archivos de código fuente (archivos .java). En este caso definimos una serie de subcarpetas para ordenar las clases en función de su utilidad, se puede ver en la siguiente figura:



  - **Network**

  Dentro de network está la clase **LoadData** que servirá para realizar todas y cada una de las llamadas tanto al servidor de Codeeta para obtener los widgets de formulario como al de Wordpress para extraer toda la información. Esta clase también contiene el método para el envió de las respuestas a los formularios de la aplicación.

  Tenemos por ejemplo el método **getWidgetElements** al que pasamos la información de una url concreta haremos peticiones y obtendremos respuestas HTTP de la siguiente

manera:

```
public boolean getWidgetElements(Context ctx, String url, String valor){
    boolean bReturn = false;
    try{
        HttpUriRequest request = new HttpGet(url);
        HttpResponse resp = mHttpClient.execute(request);
        InputStream input = resp.getEntity().getContent();
        String jString = PWUtilities.convertStreamToString(input);
        if (jString.length() != 0){
            storeWidget(ctx, jString, valor);
            bReturn = true;
        }else{
            bReturn = false;
        }
        input.close();
    }catch(Exception e){
        bReturn = false;
    }

    return bReturn;
}
```

Con este método obtenemos el String jString con toda la información de la llamada realizada bien sea un widget de codeeta o la información de cualquiera de las secciones de la página. Este String realmente es el contenido de una respuesta JSON que posteriormente podremos parsear para obtener los valores deseados.

- **Parsing**

Este carpeta únicamente contiene la clase **JSONParser** que se encarga mediante su método **getJSONFromUrl** de dada una url devolver el objeto JSON correspondiente a esa petición.

- **Ui**

Contiene la parte más importante del desarrollo, utiliza las clases de las otras tres carpetas para definir todo el funcionamiento de la aplicación.

La estructura de las clases es muy similar, se utiliza el código de la clase BeneficiosActivity como muestra:

- Importamos todas las clases necesarias para la implementación.
- Declaramos la clase implementando las interfaces: View.OnClickListener, View.OnTouchListener para controlar los diferentes toques del usuario sobre la pantalla para cambiar el color de las opciones seleccionadas, asi como controlar el movimiento de unas pantallas a otras.

```
public class BeneficiosActivity extends Activity
implements View.OnClickListener, View.OnTouchListener {
```

- Lo primero que implementamos es el método onCreate. Este método se debe implementa en toda Activity, siendo invocado siempre que se inicia la actividad. Este método recibe como parámetro un objeto de la clase *Bundle*, que contiene el estado anterior de la actividad en caso de que haya sido suspendida. En este método indicamos el layout que define esta pantalla asi como inicializamos todos los componentes de la interfaz.
  Posteriormente con la clase Calendar se obtiene el momento actual para saber cuanto tiempo pasó desde la última actualización del contenido y en caso de ser superior a un día se vuelve a actualizar.

```
public void onCreate(Bundle savedInstanceState) {

    super.onCreate(savedInstanceState);
    setContentView(R.layout.webview);

     inicializarComponentes();

        Calendar calendario = Calendar.getInstance();
        long tiempo_actual =calendario.getTimeInMillis();

        prefs = getSharedPreferences(Constants.PREFERENCES_NAME,
                Context.MODE_PRIVATE);

        long contador = prefs.getLong("contador_beneficios", 0);
        long diferencia_tiempo = tiempo_actual - contador;

        if(diferencia_tiempo>= Constants.DIA){
             wDialog = new ProgressDialog(BeneficiosActivity.this);
             wDialog.setMessage
                (BeneficiosActivity.this.getString(R.string.load_progress));
             wDialog.setCancelable(false);
             wDialog.show();

             mLoadingThread = new LoadingThread(handler);
             mLoadingThread.start();
        }
        else{
            crearPagina();
        }
}
```

Para actualizar lanzamos un hilo que recuperará la información actualizada, mientras se completa este proceso se muestra un diálogo de progreso. Primero se establece la conexión y posteriormente se llama a la API JSON de Wordpress mediante la correspondiente URL. La información obtenida será guardada así como el momento en el que se ha accedido a dicha información para saber posteriormente en que momento se debe actualizar.

```
LoadData data = new LoadData(httpClient);
if(data.getElements(getApplicationContext(),
        Constants.BENEFICIOS_URL, "beneficios")){
    Calendar calendario = Calendar.getInstance();
    prefs.edit().putLong("contador_beneficios",
            calendario.getTimeInMillis()).commit();
    setState(STATE_LOADED);
}
```

Si el tiempo establecido para volver a actualizar aún no ha pasado se llama al método **crearPagina** que trata el contenido del JSON obtenido tras las correspondientes llamadas obteniendo los campos necesarios para mostrarlos posteriormente.

```
public void crearPagina(){

    try {
        beneficios=prefs.getString("beneficios", Constants.JSON_EMPTY);
        JSONObject json = new JSONObject(beneficios);
        posts = json.getJSONArray(TAG_POSTS);
        JSONObject c = posts.getJSONObject(0);

        title = c.getString(TAG_TITLE);
        content = c.getString(TAG_CONTENT);

        InterfazJavascript interfazjs= new InterfazJavascript(title,content);

        contenido.getSettings().setJavaScriptEnabled(true);
        contenido.addJavascriptInterface(interfazjs, "android");
        contenido.loadUrl("file:///android_asset/conocemf.html");

    } catch (JSONException e) {
        e.printStackTrace();
    }
}
```

Una vez que tenemos el JSONObject accedemos a su posición 0 ya que es la única al ser una página. Después simplemente con el método getString de la clase JsonObject podemos acceder a toda la información deseada del JSON en este caso el título de la página y el contenido.

En este código podemos ver implementada la persistencia para almacenar los datos a los que va accediendo el usuario para posteriormente poderlo visualizar de modo offline asi como agilizar la navegación ya la información se actualizara de forma acorde a la modificación habitual de los contenidos. Utilizamos la clase SharedPreferences para esta finalidad, que consiste en el almacenamiento mediante pares (clave, valor).

Lo primero que hacemos es importarla:

```
import android.content.SharedPreferences;
```

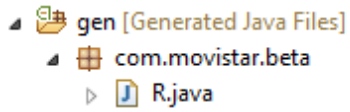
creamos una instancia de la misma y la inicializamos:

```
   private SharedPreferences prefs=
getSharedPreferences(Constants.PREFERENCES_NAME,
Context.MODE_PRIVATE);

para almacenar un valor, en este caso un String con la cadena JSON
obtenida de la correspondiente llamada:

prefs.edit().putString(value, joObject.toString()).commit();

value será el nombre asociado a lo que estemos almacenado para su
posterior recuperación. Y lo recuperaremos de la siguiente manera:

String resultado=prefs.getString(value, Constants.JSON_EMPTY);
```

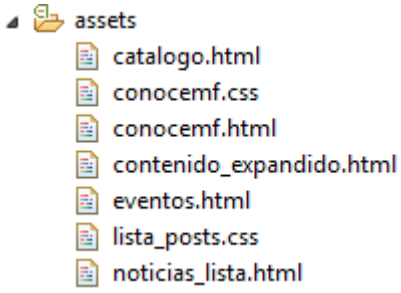
- **Util**

En esta carpeta se encuentran varias clases que definen utilidades necesarias desde diferentes clases del proyecto. Tenemos por ejemplo la clase **Constants** con todas las constantes necesarias a lo largo del proyecto. También contiene la clase **PWUtilities** con el método **convertStreamToString** que convierte un InputStream que le pasamos resultante de las diferentes llamadas en un String.

- /**gen**: aquí aparecerán archivos que genera Eclipse automáticamente, como el archivo R.java que contiene las posiciones en memoria de las variables del proyecto.



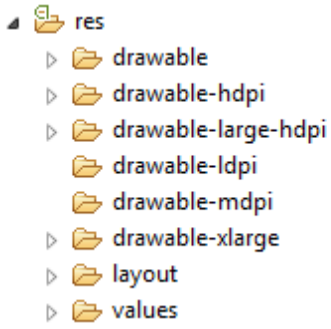
- **/assets**: es un directorio para guardar recursos que utilice la aplicación. Para acceder a los recursos de este directorio es necesario hacer uso de la clase *AssetManager* para leer los datos.



En este caso tenemos definidos una serie de archivos html y css para definir la estructura y estilo de los diversos WebViews que utilizamos en la aplicación. El contenido que

obtenemos de Wordpress y Codeeta será mostrado en el terminal móvil en forma de WebViews.

- **/res**: es el directorio donde se guardan todos los recursos que utiliza la aplicación, tanto imágenes y archivos multimedia como diversos xml que definen diferentes aspectos de la aplicación. Los recursos colocados en este directorio son fácilmente accesibles desde la clase *R*.



En las carpetas **drawable** se guardan las imágenes utilizadas en la aplicación, hay varias carpetas para ordenarlas en función del dispositivos que utilizamos por ejemplo dependiendo de si es tablet o móvil. En **layout** están los ficheros xml que definen la estructura de cada una de las pantallas de la aplicación.

Como ejemplo, podemos ver el layout que define la página de noticias:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:background="@color/white">
<include android:id="@+id/head"  layout="@layout/head"/>
<include
    android:id="@+id/home"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_below="@id/head"
    layout="@layout/home" />
<TextView
    android:id="@+id/categoria"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text=""
    android:gravity="left"
    android:textSize="40px"
    android:textColor="@color/titulo_categoria"
    android:paddingLeft="20px"
    android:layout_below="@id/home"/>
<View android:background="@color/titulo_categoria"
    android:id="@+id/lineaHorizontal"
```

```
    android:layout_height="1dp"
    android:layout_width="fill_parent"
    android:paddingLeft="10px"
    android:layout_below="@id/categoria">
</View>
 <ListView
      android:layout_above="@id/bottom"
      android:layout_below="@id/lineaHorizontal"
      android:id="@android:id/list"
      android:layout_width="fill_parent"
      android:layout_height="wrap_content"
      android:layout_gravity="center_vertical"/>
 <include android:id="@+id/bottom" layout="@layout/bottom" />
</RelativeLayout>
```

Hay varios tipos de layouts en función del tipo de distribución que queramos en la pantalla del dispositivo. En este caso tenemos un **RelativeLayout,** la mayoría de las veces utilizamos este debido a su flexibilidad, los elementos se colocan relativos a otro elementos o layouts.

Despues incluimos la imagen de la cabecera, asi como el icono de home que están en otro archivo .xml y por tanto debemos importarlos.

A continuación se añade un TextView para indicar en que sección estamos, en este caso Noticias.

View define únicamente la línea horizontal que separa el texto de título de las noticias.

La lista de noticias la definimos mediante un ListView al que se van incluyendo los posts de dicha categoría.

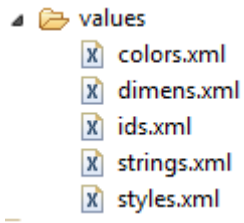
Finalmente mediante otro import añadimos un xml que define el pie de la aplicación.

Como podemos ver cada uno de estos componentes tiene una serie de propiedades para poder personalizar totalmente el contenido, propiedades como: textSize, textColor, layout_above, layout_below (indica entre que componentes esta colocado), paddingLeft, background, layout_width, layout_height, etc.

Con este layout la pantalla que se define es la siguiente:

Figura 24. Seccion Noticias en versión Android

Por último dentro de **/res** tenemos ficheros xml que definen una serie de colores, tamaños de letra, strings, stilos, etc.



- **AndroidManifest.xml**: es el archivo de configuración de nuestra aplicación. Se genera automáticamente al crear un proyecto y en él se declaran todas las especificaciones de nuestra aplicación. Cuando hablamos de especificaciones hacemos mención a las Activities utilizadas, los Intents, bibliotecas, el nombre de la aplicación, el hardware que se necesitará, los permisos de la aplicación, etc. El manifest de esta aplicación quedará de la siguiente forma:

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.movistar.beta"
    android:versionCode="4"
    android:versionName="1.3">
  <uses-sdk android:minSdkVersion="8" />
        <uses-permission android:name="android.permission.INTERNET" />
        <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
  <application android:label="@string/app_name"
     android:icon="@drawable/launcher">
        <activity android:name=".ui.InicioActivity" android:label="@string/app_name"
android:theme="@style/Theme.PuntosWifi" android:screenOrientation="portrait">
```

```
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity android:name=".ui.MenuActivity" android:label="@string/app_name"
android:theme="@android:style/Theme.NoTitleBar" android:screenOrientation="portrait"/>
                    <activity android:name=".ui.ContactoActivity"
android:theme="@android:style/Theme.NoTitleBar" android:screenOrientation="portrait"/>
                    <activity android:name=".ui.ConocemfActivity"
android:theme="@android:style/Theme.NoTitleBar" android:screenOrientation="portrait"/>
                    <activity android:name=".ui.QueesActivity"
android:theme="@android:style/Theme.NoTitleBar" android:screenOrientation="portrait"/>
                    <activity android:name=".ui.SingleMenuItemActivity"
android:theme="@android:style/Theme.NoTitleBar" android:screenOrientation="portrait"/>
                    <activity android:name=".ui.SingleMenuItemActivity_Catalogo"
android:theme="@android:style/Theme.NoTitleBar" android:screenOrientation="portrait"/>
                    <activity android:name=".ui.BeneficiosActivity"
android:theme="@android:style/Theme.NoTitleBar" android:screenOrientation="portrait"/>
                    <activity android:name=".ui.CasosExitoActivity"
android:theme="@android:style/Theme.NoTitleBar" android:screenOrientation="portrait"/>
                    <activity android:name=".ui.CatalogoActivity"
android:theme="@android:style/Theme.NoTitleBar" android:screenOrientation="portrait"/>
                    <activity android:name=".ui.NoticiasActivity2"
android:theme="@android:style/Theme.NoTitleBar" android:screenOrientation="portrait"/>

                    <activity android:name=".ui.RecomendarActivity"
android:theme="@android:style/Theme.NoTitleBar" android:screenOrientation="portrait"/>
                    <activity android:name=".ui.RRSSActivity"
android:theme="@android:style/Theme.NoTitleBar" android:screenOrientation="portrait"/>
                    <activity android:name=".ui.SolucionesActivity"
android:theme="@android:style/Theme.NoTitleBar" android:screenOrientation="portrait"/>
                    <activity android:name=".ui.UneteActivity"
android:theme="@android:style/Theme.NoTitleBar" android:screenOrientation="portrait"/>
                    <activity android:name=".ui.NoticiasActivity"
android:theme="@android:style/Theme.NoTitleBar" android:screenOrientation="portrait"/>
                    <activity android:name=".ui.EventosActivity"
android:theme="@android:style/Theme.NoTitleBar" android:screenOrientation="portrait"/>
                    <activity android:name=".ui.Soluciones_FuerzasVentaActivity"
android:theme="@android:style/Theme.NoTitleBar" android:screenOrientation="portrait"/>
                    <activity android:name=".ui.SingleMenuActivity"
android:theme="@android:style/Theme.NoTitleBar" android:screenOrientation="portrait"/>
                    <activity android:name=".ui.PruebaCatalogo"
android:theme="@android:style/Theme.NoTitleBar" android:screenOrientation="portrait"/>
                    <activity android:name=".ui.Soluciones_FuerzasCampo"
android:theme="@android:style/Theme.NoTitleBar" android:screenOrientation="portrait"/>
                    <activity android:name=".ui.Soluciones_Flotas"
android:theme="@android:style/Theme.NoTitleBar" android:screenOrientation="portrait"/>
                    <activity android:name=".ui.Soluciones_MarketingMovil"
android:theme="@android:style/Theme.NoTitleBar" android:screenOrientation="portrait"/>
                    <activity android:name=".ui.Soluciones_M2M_Domotica_Telecontrol"
android:theme="@android:style/Theme.NoTitleBar" android:screenOrientation="portrait"/>
                    <activity android:name=".ui.CasosExito_AdmPublica"
android:theme="@android:style/Theme.NoTitleBar" android:screenOrientation="portrait"/>
                    <activity android:name=".ui.CasosExito_Aseguradoras"
android:theme="@android:style/Theme.NoTitleBar" android:screenOrientation="portrait"/>
                    <activity android:name=".ui.CasosExito_Automocion"
android:theme="@android:style/Theme.NoTitleBar" android:screenOrientation="portrait"/>
                    <activity android:name=".ui.CasosExito_BancosCajas"
android:theme="@android:style/Theme.NoTitleBar" android:screenOrientation="portrait"/>
```

```
                <activity android:name=".ui.CasosExito_Construccion"
android:theme="@android:style/Theme.NoTitleBar" android:screenOrientation="portrait"/>
                <activity android:name=".ui.CasosExito_Cultura"
android:theme="@android:style/Theme.NoTitleBar" android:screenOrientation="portrait"/>
                <activity android:name=".ui.CasosExito_DistrTransLog"
android:theme="@android:style/Theme.NoTitleBar" android:screenOrientation="portrait"/>
                <activity android:name=".ui.CasosExito_Educacion"
android:theme="@android:style/Theme.NoTitleBar" android:screenOrientation="portrait"/>
                <activity android:name=".ui.CasosExito_Energia"
android:theme="@android:style/Theme.NoTitleBar" android:screenOrientation="portrait"/>
                <activity android:name=".ui.CasosExito_MediaPublicidad"
android:theme="@android:style/Theme.NoTitleBar" android:screenOrientation="portrait"/>
                <activity android:name=".ui.CasosExito_OcioTurismo"
android:theme="@android:style/Theme.NoTitleBar" android:screenOrientation="portrait"/>
                <activity android:name=".ui.CasosExito_SanidadFarma"
android:theme="@android:style/Theme.NoTitleBar" android:screenOrientation="portrait"/>
                <activity android:name=".ui.CasosExito_Seguridad"
android:theme="@android:style/Theme.NoTitleBar" android:screenOrientation="portrait"/>
                <activity android:name=".ui.CasosExito_Tramites"
android:theme="@android:style/Theme.NoTitleBar" android:screenOrientation="portrait"/>
                <activity android:name=".ui.NoticiasEventos"
android:theme="@android:style/Theme.NoTitleBar" android:screenOrientation="portrait"/>
    </application>
</manifest>
```

A continuación se explican los puntos más importantes de este fichero:

- Nombre de la aplicación e icono de la aplicación al instalarla:

```
<application android:label="@string/app_name"
    android:icon="@drawable/launcher">
```

  En este caso estamos haciendo referencia a otras ubicaciones donde tenemos el app_name que es “Movilforum” y la imagen de lanzamiento de la aplicación. Tanto string como drawable son directorios dentro de res donde están todos los recursos.

- Declara el **nivel mínimo del API Android, la SDK 8 corresponde a la versión 2.2 de Android.**

```
<uses-sdk android:minSdkVersion="8" />
```

- Se define también el nombre del paquete Java, que sirve como identificado único de la aplicación. Este nombre en este caso es com.movistar.movilforum.
  En la siguiente línea vemos el versionCode, que hacer referencia al número de versión de desarrollo de la aplicación. Cada versión final que se desea publicar debe tener un valor diferente.

  El versionName es el número de versión del programa.

  En este caso son 4 y 1.3 respectivamentes debido que por reformas y diferentes cambios en la aplicación ha sido necesario ir generando nuevas versiones, estando actualmente en la 4ª.

```
<manifest
xmlns:android="http://schemas.android.com/apk/res/android"
      package="com.movistar.movilforum"
      android:versionCode="4"
     android:versionName="1.3">
```

Mediante este Tag especificamos los permisos que va a necesitar nuestra aplicación para poder ejecutarse, además son los que deberá aceptar el usuario antes de instalarla.

Chequear si el dispositivo móvil dispone de una **conexión a internet**. Muchos de los fallos en aplicaciones para Android vienen a causa de no tratar todos los casos posibles. Para una aplicación que requiera contenido de la red de redes, lo correcto es comenzar verificando que el dispositivo tiene conexión a internet y en caso de no teneral responder de una forma adecuada.

```
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"
/>
```

Mediante el filtro <intent-filter> definimos el punto de entrada a la aplicación que será InicioActivity en este caso.

```
<activity android:name=".ui.InicioActivity"
android:label="@string/app_name" android:screenOrientation="portrait">
     <intent-filter>
          <action android:name="android.intent.action.MAIN" />
          <category android:name="android.intent.category.LAUNCHER" />
     </intent-filter>
 </activity>
```

Finalmente debemos de listar todas las Actividades que compondrán la aplicación, lo hacemos de la siguiente forma:

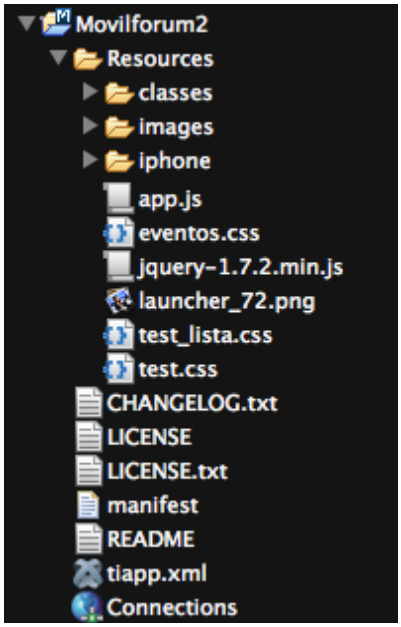
```
<activity android:name=".ui.MenuActivity" android:label="@string/app_name"
android:theme="@android:style/Theme.NoTitleBar" android:screenOrientation="portrait"/>
```

Definimos el nombre de la actividad, el de la aplicación, un tema(en este caso solo omitimos la barra de título que por defecto se incluye) y por último indicamos que la aplicación solo funcionara en orientación vertical. De este mismo modo definiremos todas las demás actividades.

- **default.properties**: es un fichero que genera automáticamente Eclipse.

5.2 Implementación aplicación iOS:

La estructura del proyecto en Titanium es la siguiente:



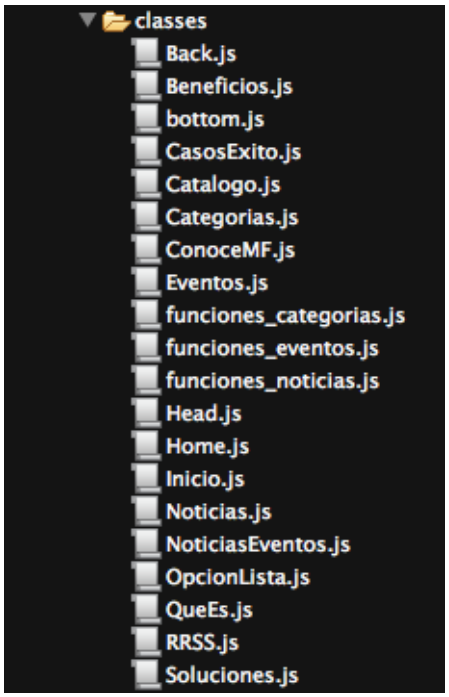
Los tres puntos fundamentales de esta estructura son:

Build: en este folder es donde titanium genera el código nativo de la aplicación, en este caso los archivos en Ojective-C. Titanium lo utiliza para la compilación del proyecto pero el usuario no tiene que hacer nada en el mismo. Dentro de esta carpeta tenemos la carpeta iphone donde encontraremos el archivo *.xcodeproj que abre el proyecto en Xcode con el código ya convertido a Objective-C.

- **Resources**: en esta carpeta se guardan todo los archivos del código javaScript, imágenes, css, html, etc.

Dentro de esta carpeta se han clasificado los archivos en varias subcarpetas:

- Clases: aquí se encuentran todos los JavaScript que definen la aplicación:



Destacamos a continuación algunos aspectos del desarrollo:

Las clases Back.js, Head.js y Home.js son funciones que devuelven los botones de retroceso, de ir directamente a la Home y el icono superior de la aplicación.

Bottom.js define la parte inferior que será visible durante toda la navegación por la aplicación. En la parte inferior de la pantalla se encuentran tres botones: contacto, recomendar y únete. A la función bottom le pasaremos una url y un subject para la opción recomendar, ya que en cada momento se podrá recomendar por redes sociales asi como por correo la información que el usuario esta visualizando en ese momento. Para esta funcionalidad contamos con el siguiente método:

```
  function recomen(e)
{
     if(e.index==0){
    var emailDialog = Titanium.UI.createEmailDialog();
emailDialog.subject = subject;
emailDialog.toRecipients = [];
emailDialog.messageBody = url;
emailDialog.open();
 }
  if(e.index==1){
        url_t = 'http://twitter.com/intent/tweet?text='+subject+'&url='+url;
        Titanium.Platform.openURL(url_t);
 }
  if(e.index==2){
        url_f = 'https://www.facebook.com/sharer/sharer.php?u='+url+'&'+subject;
        Titanium.Platform.openURL(url_f);
 }
};

    recomendar.addEventListener('click', function(e){
dialog.show();

dialog.addEventListener('click',recomen);

  });
```

Los otros dos botones del bottom son formularios de codeeta cuya información se obtiende mediante llamadas http con respuesta JSON que se parseará para mostrar un formulario con los mismos datos pero adaptado al dispositivo. Una de las llamadas más importantes es **get_widget_elements**, en el caso de querer obtener un widget de Codeeta se define de la siguiente manera:

```
get_widget_elements = function(callback, url_widget, widget_id, tipo){
    if(connection){
        server_call(url_widget, 'GET', null, callback, 'json');

    }else{
        alert("El formulario no ha podido actualizarse porque no existe conexión a Internet");
        getWidget(widget_id, tipo);

    }
};
```

En caso de estar llamando a Wordpress será de la siguiente forma:

```
get_widget_elements = function(callback){

    var d_actual = new Date();
        var t_actual = d_actual.getTime();
        contador= Ti.App.Properties.getDouble('contador_beneficios');
        tiempo_actualizar=t_actual-contador>86400000;
        var ben=Ti.App.Properties.getString('beneficios')
    if(connection){
        if(tiempo_actualizar){

        view_loading.open();
        server_call(url, 'GET', null, callback, 'json');
        }
        else {

            crearPagina(ben);

        }
    }else{
        alert("Sin conexión de red");
        if(ben!='{}'||ben!=null){crearPagina(ben);}

    }
};
```

El proceso es muy similar solo que bastará con una url sin necesidad de más parámetros adicionales.

El método server_call al que se refieren los anteriores se define como sigue:

Si hay conexión dados

```
server_call = function(url, type, options, callback, responseType){
    //send to codeeta
    var xhr = Titanium.Network.createHTTPClient();
    xhr.open(type, url);
    if(responseType == 'json'){
        xhr.setRequestHeader('Accept', 'application/json');
    }
    xhr.onerror = function(e) {
        if(connection){
            alert('Error de comunicación');
        }
        return false;
    };
    xhr.onload = function() {
        if(responseType == 'xml'){
            test = this.responseXML;
        }else if(responseType === 'json'){
            test = this.responseText;
        }
        callback.call(this, test);
    };
    if(options != null){
        xhr.send(options);
    }else{
        xhr.send();
    }
};
```

Definiendo un cliente HTTP se realiza la conexión dada la url correspondiente y se obtiene toda la información requerida.

Finalmente habiendo obtenido ya la información solo queda parsear el JSON que la contiene y mostrarla mediante un WebView:

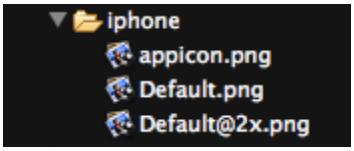
```
crearPagina = function(test){

        jsonResponse = JSON.parse(test);
        array=jsonResponse.posts;
        beneficios= (array[0].content);
        view_loading.close();
         webview_beneficios.html=
         '<head><link rel="stylesheet" type="text/css" href="../test.css"/></head>'+
         '<body><div>'+beneficios+'<div><body>';

}
```

Al igual que en la versión para Android se ha dotado a la aplicación para iOS de persistencia, se hace mediante la clase Properties, igualmente se definen pares clave valor que almacenan la cadena JSON dándole una clave para su posterior recuperación:

```
Ti.App.Properties.setString(valor, test);

var ben=Ti.App.Properties.getString(valor)
```

- Images: almacena todas las imágenes que incluye la aplicación en formato .png.

- Iphone: incluye las imágenes para lanzar la aplicación, tanto la primera que se mostrará al cargar la aplicación (Default.png para iPhone y Default@2x.png para iPad) así como el icono que la representará en el menú del dispositivo sobre el que se ejecuta(appicon.png).



Por último podemos ver una serie de archivos sueltos como app.js que será el primer .js que se cargará, algunos css que definirán el estilo de los WebViews así como la librería de jquery que utilizaremos para parsear el contenido de los WebViews al igual que hicimos en Android.

El último fichero a destacar es:

- **tiapp.xml** : es el archivo de configuracion de la aplicación, contiene todas la propiedades de la misma

```
<?xml version="1.0" encoding="UTF-8"?>

<ti:app xmlns:ti="http://ti.appcelerator.org">

   <property name="acs-oauth-secret-
production" type="string">EOJrz2AXLeucl0TBf2LDiLqbhDCPKjfZ</property>

   <property name="acs-oauth-key-
production" type="string">4WNVs4pmrqvj1nk8XcieQklbr1WOM95L</property>

   <property name="acs-api-key-
production" type="string">6vOOtprHUIjectsVx0OCT6o6RO980qnB</property>

   <property name="acs-oauth-secret-
development" type="string">dE6WECT9cie0ratsHp6xMbBvbnbsDrnr</property>

   <property name="acs-oauth-key-
development" type="string">PEojEClcAkD834RVbqzeBNUCWyDrW0h8</property>

   <property name="acs-api-key-
development" type="string">anwg2E4U6VHZcn0mmYZjLQk9BZhS9oaN</property>

   <deployment-targets>

      <target device="mobileweb">false</target>

      <target device="iphone">true</target>
```

```
    <target device="ipad">true</target>
    <target device="android">false</target>
    <target device="blackberry">false</target>
</deployment-targets>
<sdk-version>2.0.1.GA2</sdk-version>
<id>com.tyven.movilforum</id>
<name>Movilforum</name>
<version>1.0</version>
<publisher>patricia</publisher>
<url>http://</url>
<description>not specified</description>
<copyright>2012 by patricia</copyright>
<icon>appicon.png</icon>
<persistent-wifi>false</persistent-wifi>
<prerendered-icon>false</prerendered-icon>
<statusbar-style>default</statusbar-style>
<statusbar-hidden>false</statusbar-hidden>
<fullscreen>false</fullscreen>
<navbar-hidden>false</navbar-hidden>
<analytics>true</analytics>
<guid>b78d31ad-5b41-4971-88b8-ac2acca0230f</guid>
<property name="ti.ui.defaultunit">system</property>
<iphone>
    <orientations device="iphone">
        <orientation>Ti.UI.PORTRAIT</orientation>
```

```
        </orientations>
        <orientations device="ipad">
            <orientation>Ti.UI.PORTRAIT</orientation>
        </orientations>
    </iphone>
    <android xmlns:android="http://schemas.android.com/apk/res/android"/>
    <mobileweb>
        <precache/>
        <splash>
            <enabled>true</enabled>
            <inline-css-images>true</inline-css-images>
        </splash>
        <theme>default</theme>
    </mobileweb>
    <modules>
        <module platform="commonjs" version="2.0.1">ti.cloud</module>
    </modules>
</ti:app>
```

.

Al principio del fichero vemos una serie de property's que incluye el propio titanium. A continuación tenemos varios elementos con la etiqueta "<target device…" que definen para que plataformas se desarolla la aplicación, en este caso como se puede ver estarán a true iphone e ipad. También se incluyen en este archivo la versión del sdk, el identificador de la aplicación, el nombre, la versión (necesario cambiarla cada vez que subamos una actualización al App Store), quién la publica, una descripción de la misma, la orientación de la pantalla para la que esta hecha la aplicación (en este caso únicamente vertical tanto para iphone como para ios) el icono de arranque, etc.

6. Resultado

En este apartado se ve gráficamente el resultado de todo el proceso de realización del proyecto, en el lado izquierdo se encuentran las capturas de la versión Android y en el derecho las de iOS, en ambos casos se han obtenido de los respectivos emuladores. En primer lugar la pantalla de carga de la aplicación:

Posteriormente se accede al menú principal:

El menú de la parte inferior será estático para las diferentes pantallas, en el caso de Contacto y Unete tenemos formularios de Codeeta que visualizaremos de la siguiente manera (solo se muestra el caso de Contacto ya que son muy similares):

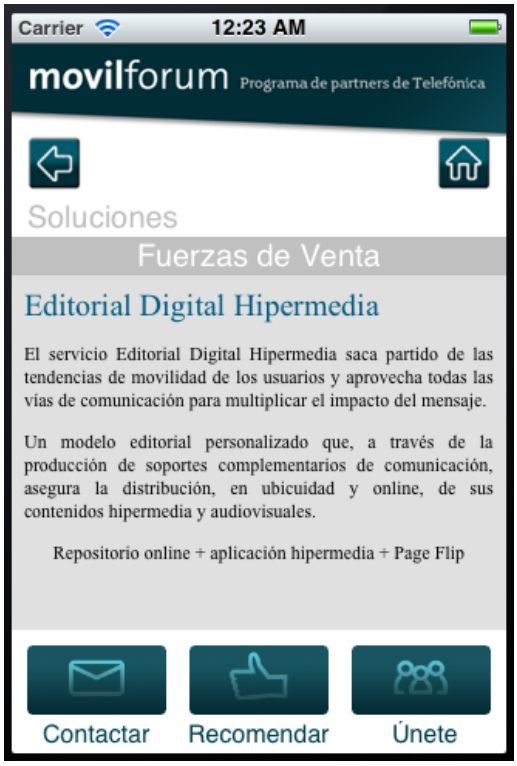
Dentro del menú inferior también se incluye la opción recomendar para recomendar el contenido que se esta visualizando en ese momento a través del correo y las redes sociales:

El primer punto del menú es Catálogo y al entrar en e mismo vamos a la siguiente pantalla:

Dado que Soluciones y Casos de Éxito están estructurados de la misma forma se muestra como ejemplo el caso de Soluciones, al pulsar en Soluciones se abre el siguiente diálogo para seleccionar la categoría deseada:

Al seleccionar la categoría por ejemplo “Fuerzas de venta” se muestra una lista de las soluciones para esta categoría

Si se quiere obtener más información acerca de los mismos al pulsar en ver más se accede a una nueva pantalla con el texto de la solución seleccionada expandido, como ejemplo para “Editorial Digital Hipermedia”:



El segundo punto del menú es el blog que se referencia de forma externa a una versión adaptada también a dispositivos móviles, viéndose de la siguiente forma:

El tercer punto del menú es Noticias y Eventos:

La visualización de las noticias es idéntica al apartado "Soluciones", mostrándose un listado de noticias y posteriormente pudiendo seleccionar cualquiera de ellas para visualizar el contenido completo de la misma. Para los eventos se muestra únicamente el nombre, fecha y lugar pudiendo igualmente pulsar sobre los mismos para obtener la información completa:

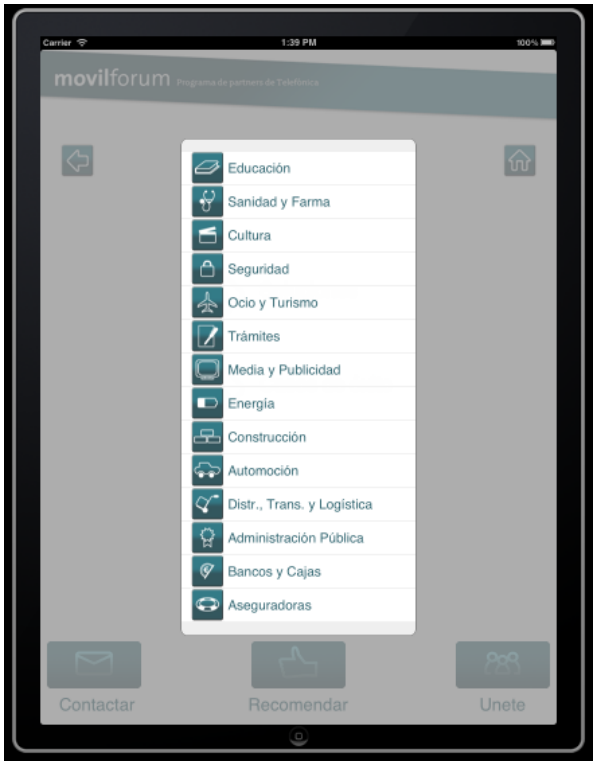
El último punto de menu es "Conoce MF" que incluye el siguiente submenu:

Las opciones “Qué es” y “Beneficios” muestran la información de manera idéntica por lo que como ejemplo se muestra el caso de “Qué es”:

Por último es posible ver el perfil de “Movilforum” en las siguientes redes sociales:

Como se ha podido comprobar ambas aplicaciones son muy similares, diferenciándose por el aspecto de los componentes propios de cada una, los colores y como matiz destacable la opción “Back” únicamente existente para iOS dado que los dispositivos Android cuentan directamente con un botón que hace esta función.

En el desarrollo para ambas plataformas se ha adaptado la aplicación para la correcta visualización en Tablets, vemos alguna imagen para el caso del Ipad:



En esta sección se hará un desglose del coste de los elementos necesarios en este proyecto. Dichos elementos incluyen costes de personal, de hardware, de software y de distribución.

El proyecto ha sido realizado entre el 20 de Enero de 2012 al 20 de Junio de 2012 por lo tanto han sido 6 meses de trabajo. Teniendo en cuenta una jornada laboral de 8 horas tendremos un total de 856 horas de trabajo, distribuidas entre diferentes tareas y diferentes roles profesionales que las llevan a cabo.

Coste de personal

Fase	Categoría	Horas	Coste/hora	Coste
Documentación	Analista	80	40€	3200€
Análisis	Analista	100	40€	4000€
Diseño	Diseñador	160	30€	4800€
Diseño gráfico	Diseñador gráfico	60	30€	1800€
Implementación	Programador	396	20€	7920€
Pruebas	Programador	60	20€	1200€
Total		**856**		**22920€**

Tabla 27. Coste de personal

Costes de hardware

Para el equipo calcularemos el coste según el periodo de amortización teniendo en cuenta una duración del proyecto de 6 meses. El equipo esta formado por ordenadores, móviles y tablets, necesitando uno de cada uno de estos dispositivos para el desarrollo Android y otro para el desarrollo en iOS.

Equipo	Coste	Coste amortizado
Samsung RC510	600€	75€
Sony Ericcson Xperia Neo V	250€	50€
Iphone 3GS	400€	80€
Mac OS X	1200€	150€
Ipad 2	520€	100€
Samsung Galaxy Note	410€	85€
Total	**3380€**	**540€**

Tabla 28. Coste de hardware

Costes de software

En la realización de este proyecto se ha optado por utilizar software libre por lo que no tenemos ningún coste asociado al hardware.

Coste distribución

Debemos pagar una cantidad por las licencias para subir la aplicación al Google Play y al App Store, en el primer caso no es necesaria la licencia para desarrollar la aplicación pero si para subirla al Google Play, en el segundo caso es necesaria tanto para el desarrollo e instalación en nuestro terminal como para su posterior distribución en el App Store.

Plataforma distribución	Coste licencia
Google Play	18€
App Store	80€
Total	**98€**

Tabla 29. Coste de distribución

Coste total

Tipo de coste	Total
Coste de personal	22920€
Coste de hardware	540€
Coste de software	0€
Coste de distribución	98€
Total	**23558€**

Tabla 30. Coste total

Por tanto el coste total del este proyecto asciende a: 23558€

Una vez concluido el proyecto y logrados los objetivos propuestos al inicio del mismo en este capitulo se presentan las conclusiones obtenidas de su desarrollo así como las posibles líneas futuras para su ampliación.

8.1 Conclusión

En la situación actual en que nos encontramos podemos suponer un futuro más centrado en los dispositivos móviles que en los propios ordenadores. Vemos que el alcance tanto de Android como de iOS es espectacular y está en constante crecimiento sorprendiéndonos cada día. Tras analizar todos los sistemas operativos móviles que hay en la actualidad se concluyó que estos dos son los que mayor franja de mercado abarcan y con muchas diferencia hoy por hoy aunque otros sigan reinventándose para conseguir hacerles sombra.

Uno de los puntos más importantes visto en este proyecto son las diferentes posibilidades a la hora de enfrentarnos a un dispositivo móvil: aplicación móvil nativa, aplicación híbrida y aplicación web HTML5. Se ha podido comprobar que el control sobre el dispositivo es la mayor ventaja de las aplicaciones nativas para dar la mejor experiencia de usuario. Es importante en todos los casos hacer un buen diseño lo más usable e intuitivo posible, buscando el atractivo y el menor número de clics para probar toda su funcionalidad ya que de otra forma el usuario puede perderse en la navegación y perder interés en la aplicación. Para determinado tipo de aplicaciones es interesante la opción hibrida como en este caso, que al ser una aplicación sobretodo de intercambio de datos que no utiliza demasiadas funcionalidades del teléfono, hemos conseguido un buen resultado. El desarrollo con Titanium supone un ahorro en recursos y una sencillez de desarrollo a tener en cuenta en términos económicos y temporales, ya que es más fácil desarrollar con HTML, JavaScript que CSS que aprender objective C y además el diseño hecho con Titanium será extrapolable a otras plataformas móviles.

8.2 Líneas futuras

Lo más interesante de esta aplicación de cara a un futuro es extrapolar esta solución para que cualquier sitio web desarrollado con Wordpress pudiera de una forma cómoda tener una adaptación de su portal al dispositivo móvil con los contenidos más relevantes del mismo. De esta forma libramos a la aplicación del peso de la gestión de todo el contenido y creamos aplicaciones muy ligeras que únicamente deben acceder mediante un protocolo sencillo de comunicación como JSON a la información seleccionada que estará actualizada en todo momento. Resultará también de utilidad y sencillez añadir formularios a la aplicación y vincularla a las diferentes redes sociales, clave para la difusión de información hoy en día.

También sería interesante en un futuro ir añadiendo más funcionalidad a la aplicación, algunas posibilidades podrían ser:

- Permitir al usuario loguearse en caso de pertenecer a Movilforum permitiéndole así no solo visualizar todos los contenidos públicos sino también otros según su restricción.
- Incluir notificaciones cada vez que se actualice algún contenido o se desee transmitir un mensaje a los usuarios de la aplicación.
- Posibilidad edición e introducción de nuevos posts y respuestas desde el dispositivo móvil. Existen plugins para conseguir esta funcionalidad.
- Únicamente se ha desarrollado la versión en español de la aplicación por lo que sería interesante dado que Movilforum tiene presencia internacional adaptarla para estos países.
- Otra posibilidad sería cubrir otras plataformas móviles, en este caso por ejemplo sería fácil llevarla a Blackberry haciendo uso del desarrollo ya implementado en Titanium para iOS.

9. Referencias

[1]. Definición dispositivo móvil Wikipedia: http://es.wikipedia.org/wiki/Dispositivo_m%C3%B3vil

[2]. Historia dispositivos móviles: http://www.cheapdslmodem.org/mobile-phone.html/

[3]. Ventas smartphone frente a móviles tradicionales: http://www.elpais.com/articulo/tecnologia/Europa/venden/smartphones/moviles/elpeputec/20110913elpeputec_3/Tes

[4]. Smartphone Market Review: http://www.gartner.com/it/page.jsp?id=1764714

[5]. Apple. Página oficial: http://www.apple.com

[6]. Apple. Comunidad desarrolladores: http://developer.apple.com

[7]. Apple. Arquitectura: http://www.techotopia.com/index.php/IPhone_iOS_5_Architecture_and_SDK_Frameworks

[8]. Android. Página oficial: http://www.android.com/

[9]. Android. Comunidad desarrolladores: http://developer.android.com/index.html

[10]. Android. Versiones: http://developer.android.com/about/dashboards/index.html

[11]. Android. Fragmentación: http://www.bgr.com/2012/05/16/android-fragmentation-visualized-opensignalmaps/

[12]. Android. Comunidad desarrolladores España: http://www.android-spa.com/

[13]. Comunidad y tutoriales Android: http://www.anddev.org/

[14]. Android : Android Apps with Eclipse, Onur Cinar, 2012. http://proquest.safaribooksonline.com/book/-/9781430244349/chapter-2-application-architecture/navpoint-15

[15]. Web HTC G1, primer dispositivo comercial con Android. http://www.htc.com/www/product/g1/overview.html

[16]. Wordpress: http://es.wordpress.org/

[17]. WordPress 3: desarrollo de proyectos Web
Wallace, Heather R. 2011

[18]. Ejemplo web Wordpress: http://www.serj.ca/

[19]. Codeeta: http://codeeta.com/

[20]. JSON: http://www.json.org/

[21]. XML-RPC: http://xmlrpc.scripting.com/spec.html

[22]. RPC. http://wiki.tools.ietf.org/html/rfc1831

[23]. RPC. http://wiki.tools.ietf.org/html/rfc1831

[24]. Appcelerator: http://www.appcelerator.com/

[25]. Xcode: https://developer.apple.com/xcode/

[26]. Comparision of cross-platform mobile development tools: http://www.idt.mdh.se/kurser/ct3340/ht12/MINICONFERENCE/FinalPapers/ircse11_submission_16.pdf

[27]. Eclipse: http://www.eclipse.org/

[28]. Mobile application development: web vs native http://cacm.acm.org/magazines/2011/5/107700-mobile-application-development/fulltext

[29]. Comparativa HTML5 vs aplicaciones nativas: http://publications.theseus.fi/bitstream/handle/10024/43719/120516_OIo05_Thesis.pdf?sequence=1

[30]. PhoneGap: http://phonegap.com/

[31]. Blackberry: http://es.blackberry.com/

[32]. Blackberry App World: http://www.blackberry.com/select/appworld/

[33]. RIM:http://www.rim.com/

[34]. Symbian Architecture: http://www.developer.nokia.com/Community/Wiki/Symbian_OS_Communication_Architecture

[35]. Nokia OVI: http://store.ovi.com/

[36]. Windows Phone 7: http://www.microsoft.com/windowsphone/es-es/default.aspx

[37]. Marketsplace: http://www.windowsphone.com/es-ES/marketplace

[38]. IDC comparativa mercado SO móviles: http://www.idc.com/getdoc.jsp?containerId=prUS23503312

[39]. Smartphone statistics and market share: http://www.email-marketing-reports.com/wireless-mobile/smartphone-statistics.htm#market

[40]. Mobile SO Evolution: http://www.xcubelabs.com/evolution-of-mobile-operating-systems.php

[41]. App Store: http://www.apple.com/es/iphone/apps-for-iphone/#heroOverview

[42]. Android Market: https://play.google.com/store

[43]. JSOUP: http://jsoup.org/

[44]. Arquitectura cliente- servidor. http://ccia.ei.uvigo.es/docencia/SCS/1011/transparencias/Tema1.pdf

[45]. Arquitectura cliente- servidor. http://www.britannica.com/EBchecked/topic/1366374/client-server-architecture

[46]. Arquitectura cliente- servidor. https://www2.bc.edu/~gallaugh/research/ism95/cccsa.html

[47]. Arquitectura cliente- servidor. http://www.oracle.com/technetwork/java/index-jsp-135888.html

[48]. Java: http://www.javahispano.com

[49]. JavaScript. http://www.w3schools.com/js/

[50]. JavaScript. http://www.desarrolloweb.com/javascript/

[51]. JavaScript. http://www.webestilo.com/javascript/

[52]. Introducción a JSON en JavaScript: http://msdn.microsoft.com/es-es/library/bb299886.aspx

[53]. CSS. http://www.w3schools.com/css/

[54]. CSS. http://www.librosweb.es/css/

[55]. CSS. http://es.html.net/tutorials/css/

[56]. HTML5. http://www.w3schools.com/html5/default.asp

[57]. HTML5. http://webdesign.about.com/od/html5/a/html_5_whats_new.htm

[58]. HTML5. http://slides.html5rocks.com/#landing-slide

[59]. HTML5. http://www.w3.org/TR/html5/

[60]. Sergey's HTML5 & CSS3 quick reference. 2010

[61]. XML. http://www.w3schools.com/xml/

[62]. XML. http://www.w3pdf.com/W3cSpec/XML/2/REC-xml11-20060816.pdf

[63]. Wikipedia: http://es.wikipedia.org/

[64]. Financial Times: http://apps.ft.com

[65]. Casos de uso, Ivar Jacobson: http://www-2.dc.uba.ar/materias/isoft1/2001_2/apuntes/CasosDeUso.pdf

[66]. Movilforum: http://espana.movilforum.com/

[67]. Blog Movilforum: http://blog.movilforum.com/

[68]. The 10 principles of mobile interface design: http://www.netmagazine.com/features/10-principles-mobile-interface-design

[69]. The best on mobile interfaz design: http://www.tappgala.com/